Informatiemodelleren met MDD

Uitwerkingen van opgaven

De gegeven uitwerkingen zijn niet altijd de enig mogelijke oplossingen. Omdat we de opgaven niet volledig hebben willen 'dichttimmeren', is er vaak ruimte voor alternatieve oplossingen.
De figuur- en tabelnummering loopt door per hoofdstuk.

Uitwerkingen hoofdstuk 1

1.1 Albumtitels worden op één plaats opgeslagen, bij het unieke albumnummer. Ze liggen op die manier eenduidig vast. Voor de componistnamen ligt dit anders. Deze wijzen niet naar een klasse; ze zijn slechts een tekstwaarde bij een track. Zodra een componist vaker voorkomt, wordt deze meervoudig opgeslagen, zonder garantie op eenduidigheid. Van standaardisatie is daardoor geen sprake.

1.2a Bij de figuren 1.12b en d is er minimaal één A-object. In dit uiterste geval zijn alle 100 B-objecten met dit ene A-object geassocieerd. Bij de figuren a, c en e kan het minimum ook 0 zijn.
In de gevallen a en b is er geen maximum voor het aantal A-objecten. In de gevallen c en d zijn er maximaal 100 A-objecten. In geval e is het maximum 33.

b Minima-maxima: 0-100, 1-100, 0-100, 1-100, 0-33.

c Minima-maxima: 0-100, 100, 0-100, 100, 0-100.

1.3 De attributen Album.titel en Track.titel illustreren het nut. Beide hebben het attribuuttype Titel met datatype Varchar(100). De ontwikkelaar heeft hiermee uitgedrukt dat het om dezelfde soort titels gaat. Blijkt op zeker moment dat Varchar(100) een te klein datatype is, dan wordt dit *op één plaats* in het model veranderd, waarna langere titels voor zowel tracks als albums beschikbaar zijn.
Als de ontwikkelaar tot de conclusie komt dat albumtitels en tracktitels verschillend behandeld moeten worden, dan moet worden gekozen voor verschillende attribuuttypen.

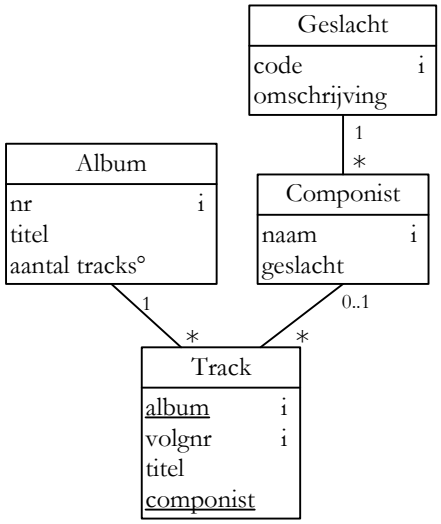
Uitwerkingen hoofdstuk 3

3.1 De beide titel-attributen zijn in het projectmodel verplicht; in het diagram is hun verplicht of optioneel zijn in het midden gelaten.

3.2a Als het attribuut geboortedatum wordt gebaseerd op een eigen attribuuttype Geboortedatum, dan zou ook een attribuut overlijdensdatum een eigen attribuuttype krijgen, namelijk Overlijdensdatum. Overlijdensdata en geboortedata zijn echter volledig vergelijkbaar en kunnen in één expressie voorkomen. Zoals in de constraint 'overlijdensdatum >= geboortedatum'. Of in de numerieke expressie 'overlijdensdatum – geboortedatum' (met als waarde een aantal dagen). Het is verstandig om vergelijkbare attributen op hetzelfde attribuuttype te definiëren, waardoor ze vanzelf hetzelfde achterliggende datatype krijgen (in dit voorbeeld: Date). Vergelijkbare kolommen in de database worden dan automatisch op hetzelfde domein gebaseerd (met datzelfde achterliggende datatype).

b Een richtlijn bij de keuze van attribuuttypen is: baseer ze op betekenis van de attribuutwaarden op zichzelf ('het zijn kalenderdata') en niet op de rollen die ze spelen in de context van de klasse ('het zijn geboortedata').

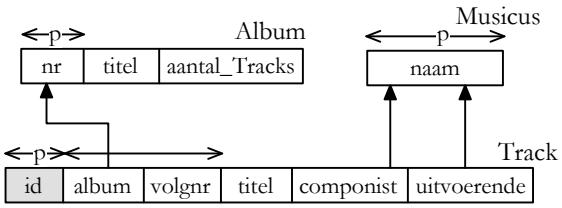
3.3 Om 'geslachten' te standaardiseren, moeten we – analoog aan de standaardisatie van componistnamen – een aparte klasse Geslacht creëren. Deze krijgt een identificerend code-attribuut met nog een tweede attribuut voor de omschrijving. Zie figuur 3.28.



Figuur 3.28 Standaardisatie van geslacht-attribuut door klasse Geslacht

Attribuut Componist.geslacht zal in de applicatie worden weergegeven via de gestandaardiseerde waarden van Geslacht.code ('m', 'v') of eventueel die van Geslacht.omschrijving ('man', 'vrouw'), zie paragraaf 3.2.2 ('Lookups in en applicatie').
Opmerking: in paragraaf 8 zullen we een andere manier zien om geslachtwaarden te beperken tot de codes (desgewenst weergegeven als een omschrijving). We modelleren daarbij geen klasse Geslacht maar maken een beperkingsregel op het attribuuttype van Componist.geslacht.

3.4 Zie het strokendiagram van figuur 3.29.

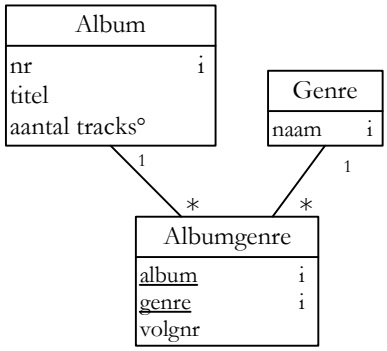


Figuur 3.29 Relationeel PSM Muziekcollectie4

3.5 Het verplicht stellen van albumgenres geeft problemen met de bestaande albums, die immers (nog) geen genre hebben. Mogelijke oplossingen zijn:
1 Als gebruiker vooraf alle genres invullen.
2 Defaults laten invullen (rechtstreeks in de database, met SQL).

3.6a Volgens figuur 3.14 horen bij elk album willekeurig veel (nul of meer) Albumgenre-objecten. Bij elk Albumgenre-object hoort precies één genre. Conclusie: bij elk album horen willekeurig veel (nul of meer) genres.

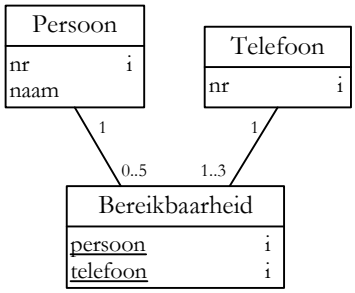
b We geven een informatiediagram met alleen Album, Albumgenre en Genre. Zie figuur 3.30.



Figuur 3.30 Muziekcollectie6 (fragment), met ordening van albumgenres d.m.v. volgnr-attribuut

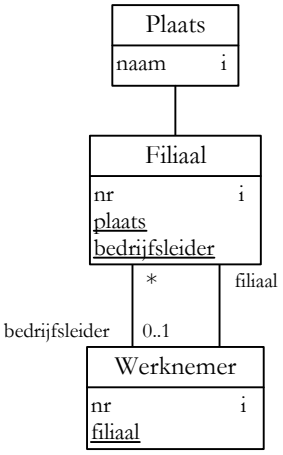
3.7a De getallen (multipliciteiten) hebben de volgende betekenis:
– Bij elke persoon horen maximaal 5 telefoons (dat wil zeggen: telefoonnummers)
– Bij elke telefoon (elk telefoonnummer) horen minimaal 1, maximaal 3 personen.

b Zie figuur 3.31.



Figuur 3.31 Completering van figuur 3.16b

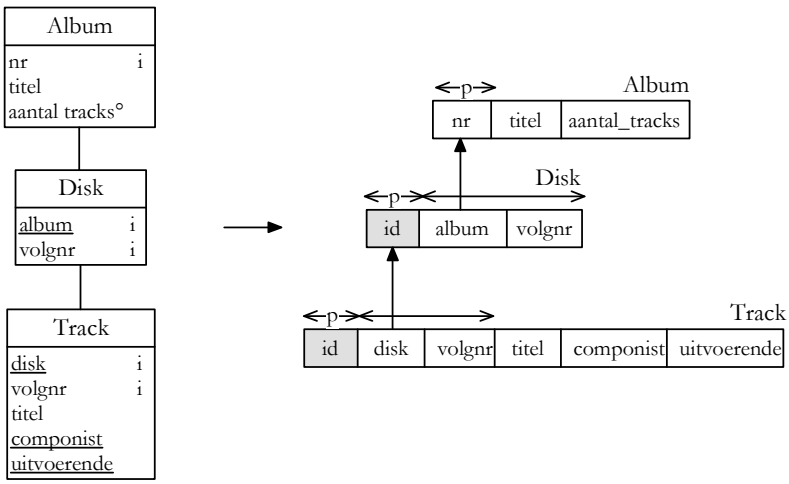
3.8 Tussen Werknemer en Filiaal bestaan twee associaties, met tegengestelde 1-op-n-richting. De tekenconventies kunnen maar voor één ervan worden gevolgd (zie figuur 3.32).



Figuur 3.32 Associaties tussen dezelfde klassen met tegengestelde 1-op-n-richting

Het diagram roept wel een aantal vragen op. Bijvoorbeeld: als een werknemer bedrijfsleider is van meerdere filialen, werkt die dan ook bij die filialen? Zo ja, dan is dat in strijd met de structuur, die immers niet meer dan één filiaal per werknemer toestaat.

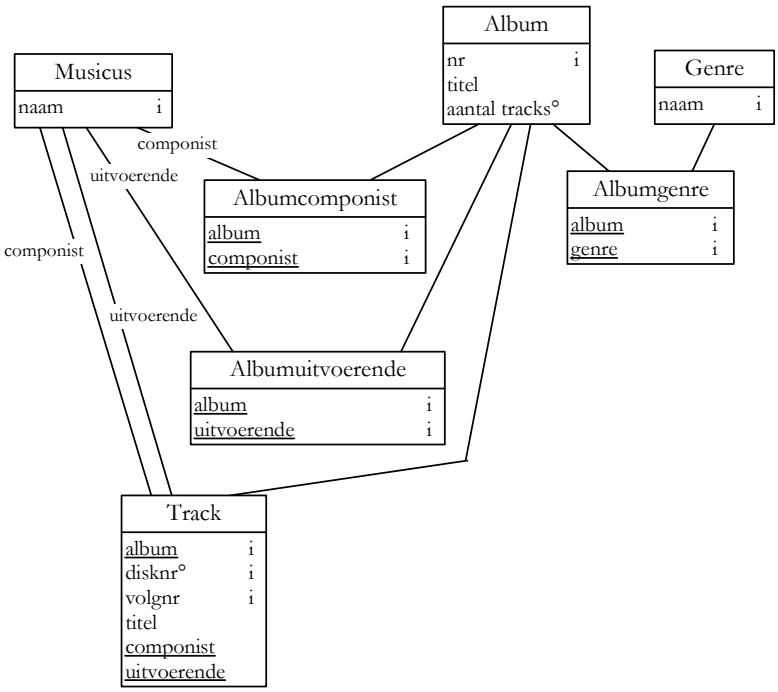
3.9 Zie figuur 3.33. Het associatieattribuut album van de klasse Disk gaat over in een verwijssleutel album van de tabel Disk. Evenzo gaat het associatieattribuut disk van de klasse Track over in een verwijssleutel disk van de tabel Track. Deze heeft kunstmatige sleutelwaarden, verwijzend naar de kunstmatige primaire sleutel Disk.id.



Figuur 3.33 Van PIM naar relationeel PSM (Muziekcollectie7, fragment)

3.10 Muziekcollectie7 is een beter uitgangspunt voor uitbreidingen waarbij extra informatie over disks moet worden vastgelegd of afgeleid. Denk bijvoorbeeld aan een speciale disknaam ('bonus-cd') of aan het (afleidbare) aantal tracks. Een disk is dan een zelfstandig 'soort van ding' met eigen eigenschappen, dat een aparte klasse verdient om die eigenschappen te kunnen vastleggen.

3.11 Zie figuur 3.34.

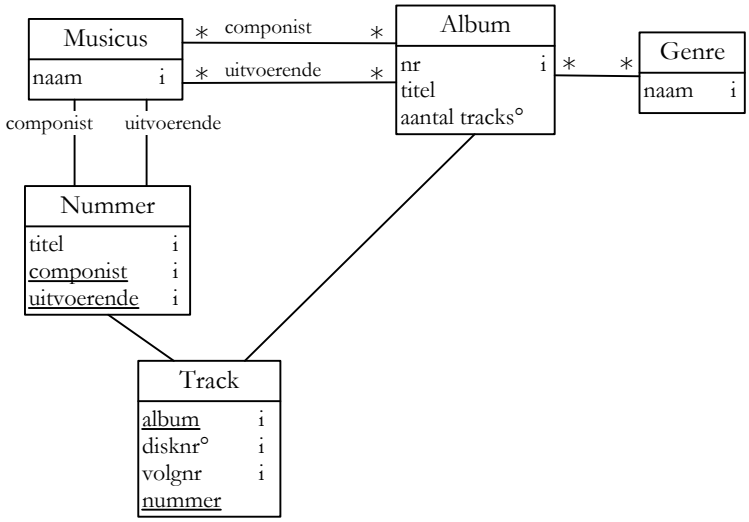


Figuur 3.34 Nogmaals Muziekcollectie10: de n-op-m-associaties uitgemodelleerd

Toelichting: zoals de associatieklasse Albumgenre staat voor alle album-genre-combinaties, zo staan de associatieklassen Albumcomponist en Albumuitvoerende voor alle album-componist- respectievelijk album-uitvoerende-combinaties.

3.12 Het nieuwe concept 'nummer' vraagt om een nieuwe klasse Nummer. Een track hoort niet alleen tot een album, maar ook tot een nummer. Omgekeerd kan een nummer gerealiseerd zijn als verschillende tracks. De nieuwe klasse Nummer wordt dus een ouderklasse van de klasse Track. De klasse Track blijft kindklasse van de klasse Album, maar wordt daarnaast kindklasse van de klasse Nummer.

De structuur wordt als in figuur 3.35.



Figuur 3.35 Muziekcollectie10 uitgebreid met klasse Nummer

Toelichting
- De attributen titel, componist en ook uitvoerende zijn overgegaan van klasse Track naar klasse Nummer.
- Track heeft een associatieattribuut nummer gekregen.
- Alleen nummers met gelijke titel, gelijke componist en gelijke uitvoerende beschouwen we als hetzelfde nummer. Een muzieknummer wordt daarom geïdentificeerd door de combinatie van titel, componist en uitvoerende.

De aanname 'Alleen nummers met gelijke titel, gelijke componist en gelijke uitvoerende beschouwen we als hetzelfde nummer' illustreert dat een informatiemodel niet alleen is gebouwd op een relevante wereld maar ook op aannamen die ertoe dienen om tot eenduidig taalgebruik te komen. Dit is een belangrijke observatie!

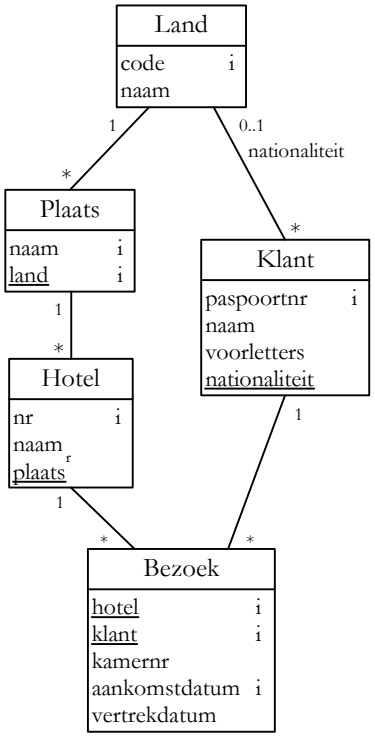
3.13a Het diagram is in meer dan een opzicht niet correct:
- Het attribuut Hotel.nr heeft een enkelvoudige i-regel en mag daarom niet optioneel zijn.
- Klasse Hotel heeft een associatie met Plaats, maar het associatieattribuut plaats is niet onderstreept.
- Associatieattribuut Hotel.plaats is verplicht (^{r}), maar de multipliciteit van de associatie is 0..1. Dit is strijdig.
- Gast.nationaliteit is een associatieattribuut, maar de associatielijn ontbreekt.
- Bezoek heeft een identificatieregel over de attributen hotel en gast. Deze combinatie is echter niet uniek: dezelfde gast kan meer dan eens een bezoek afleggen aan hetzelfde hotel. Bij één aankomstdatum is de combinatie wel uniek. Er geldt dus een identificatieregel over de drie attributen hotel, gast en aankomstdatum.

(Men kan nog twisten over de vraag of iemand op één aankomstdatum twee of meer hotels kan bezoeken. Juister gezegd: of het informatiesysteem dit moet verbieden. Zo ja, dan geldt er een identificatieregel over alleen gast en aankomstdatum.)

b Uit het diagram is niet af te leiden of vertrekdatum verplicht is. Dit wordt in het midden gelaten.

c Voorbeelden van minder contextgebonden klassenamen zijn: Klant of Relatie.

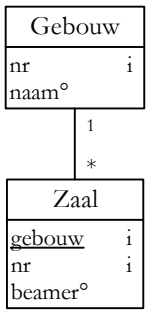
Zie figuur 3.36.



Figuur 3.36 Verbeterd informatiediagram hotelketen

3.14a Tabel 1.1 geeft informatie over twee 'soorten van dingen': zalen en gebouwen. Elke zaal bevindt zich in een gebouw, zoals een track zich bevindt op een album. We gaan uit van de volgende aannamen omtrent identificatie:
- Gebouwnr is uniek en altijd bekend en geschikt als identificatie.
- De naam van een gebouw is uniek (maar kennelijk niet altijd bekend).
- Gebouw en zaalnr zijn in combinatie identificerend voor een zaal.

Dit leidt tot twee klassen, Gebouw en Zaal. Zie figuur 3.37. De identificatieregels zijn conform de aannamen.

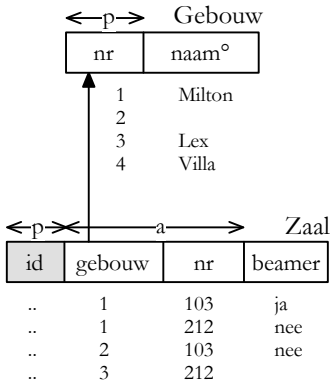


Figuur 3.37 Informatiediagram gebouwen en zalen

Omdat er maar van één soort zaalvoorziening sprake is (beamer), hebben we dit worden gemodelleerd met een attribuut Zaal.beamer, met als mogelijke waarden 'j' en 'n' (voor 'ja' en 'nee'). Omdat niet altijd bekend is of een zaal een beamer heeft, is dit attribuut optioneel.

b Aannemelijk is dat de namen van de gebouwen uniek zijn. De diagramtechniek staat echter maar één i-regel toe (primaire identificatie). In hoofdstuk 4 wordt een notatie behandeld voor aanvullende uniciteitsregels.

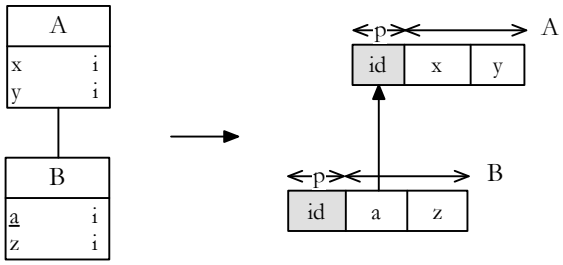
c Zie figuur 3.38.



Figuur 3.38 PSM met populatie bij figuur 3.33

3.15 Door de klasse Gebouw van figuur 3.26a (en de gelijknamige tabel van figuur 3.26b) worden gebouwnummers (evenals gebouwnamen) gestandaardiseerd. Er is daarom helemaal geen behoefte aan nog een extra klasse Gebouwnr.
Opmerking: de klassenaam Gebouwnr is op zichzelf al een signaal dat er iets niet in orde is. De naam 'Gebouwnr' is immers (afgezien van de hoofdletter) typisch een attribuutnaam (eigenschapnaam) en geen klassenaam (naam van een 'soort ding'). Een tweede aanwijzing is dat Gebouw en Gebouwnr een gelijksoortig identificerend attribuut hebben. Ten slotte is het opmerkelijk dat het enkelvoudige, identificerende attribuut Gebouw.nr ook een associatieattribuut is.

3.16 Figuur 3.39 toont hoe het informatiemodel van figuur 3.27 wordt getransformeerd tot een algemeen relationeel PSM.

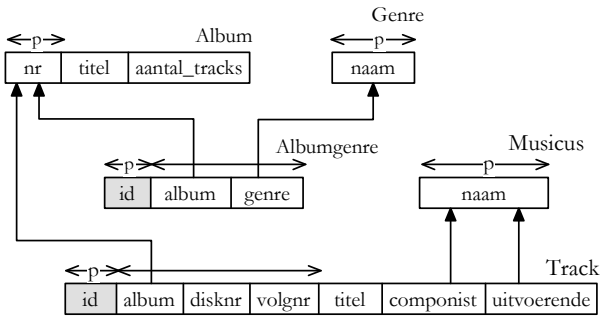


Figuur 3.39 Transformatie van PIM naar PSM

Opmerking: de structuur is die van Disk en Track uit Muziekcollectie8.

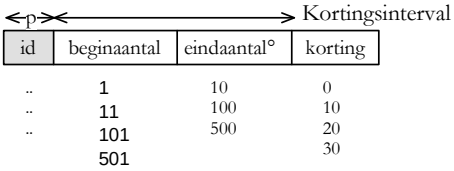
Uitwerkingen hoofdstuk 4

4.1a Zie figuur 4.21.



Figuur 4.21 Algemeen relationeel PSM van Muziekcollectie8

4.2 Figuur 4.22 geeft het gevraagde PSM met populatie. De regels zijn 'meegenomen' van het PIM naar het PSM. Merk op dat eindaantal optioneel is, zij het dat de constraint die optionaliteit slechts toestaat voor het hoogste interval.



Constraints:
- als eindaantal niet null, dan beginaantal <= eindaantal
- intervallen beginaantal-eindaantal zijn aansluitend vanaf 1 (en dus niet-overlappend)

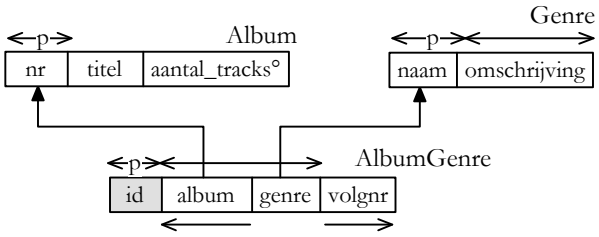
Figuur 4.22 PSM-equivalent van figuur 4.4b, met populatie

4.3 De waarden van Genre.naam zullen in het algemeen korter zijn dan van Genre.omschrijving. En ook minder veranderlijk. Dit maakt ze meer geschikt. Bedenk hierbij dat het identificerende attribuut wordt getransformeerd naar een primaire sleutel van het PSM en dat de waarden ervan ook als verwijssleutelwaarden optreden.

Verder is de omschrijving weliswaar verplicht, maar hoe stabiel is die regel? Dat is een vraag die een ontwikkelaar zich altijd moet stellen. Bij doorvragen zal misschien zelfs blijken dat een eindgebruiker alleen maar last heeft van de verplichting altijd direct een genreomschrijving te moeten invoeren. De ontwikkelaar doet er goed aan de structuur van het model (PIM en PSM) zo min mogelijk van deze eis afhankelijk te maken.

4.4a Er kunnen verschillende albums zijn die hetzelfde 'eerste genre' hebben.

b Zie figuur 4.23 voor een PSM (strokendiagram) bij figuur 4.5.



Figuur 4.23 PSM bij PIM-diagram van figuur 4.5

Opmerkingen: dit diagram toont geen verplichte waarden (evenmin als dat van figuur 4.5). De uniciteitsregel over de niet-aangrenzende kolommen album en volgnr wordt weergegeven door een onderbroken pijl.

c Omdraaien van de i's en de u's geeft hetzelfde diagram. Omdat het om allemaal verplichte kolommen gaat, vormen zowel (album, volgnr) als (album, genre) een alternatieve sleutel.

d Voor het PSM maakt het niets uit, zie onderdeel c. Een argument zouden we kunnen ontlenen aan de vraag hoe stabiel we de verplichte-waarderegels voor genre en volgnr inschatten. Een (album, genre)-combinatie zonder volgnr is goed denkbaar, een (album, volgnr)-combinatie zonder genre lijkt weinig zinvol. Dit geeft een lichte voorkeur voor de combinatie (album, genre).

4.5a De zin is afleidbaar uit die van figuur 4.10a. Dat wil zeggen: als de zinnen in die figuur als ware uitspraken gelden over de relevante wereld, dan geldt dat ook voor de gegeven zin.

b Omdat albumtitel niet uniek is, is de gegeven zin niet te herleiden tot één specifiek object. Merk op dat er staat 'een album' en niet 'één album'.

c Omdat er meerdere albums mogelijk zijn met de titel 'Really', is 'album 'Really'' geen correcte objectformulering. *Opmerking*: voor een populatie waarin die titel maar één keer voorkomt, zou dat te verdedigen zijn. Bij het modelleren gaat onze interesse echter niet uit naar toevallige populaties, maar naar de structuur. Het taalgebruik wordt daaraan getoetst, ook als het om voorbeelden gaat.

Ook ‘track 1’ is geen correcte objectformulering, omdat ‘track 1’ dubbelzinnig is. Onduidelijk is om welke track (op welk album) het gaat.

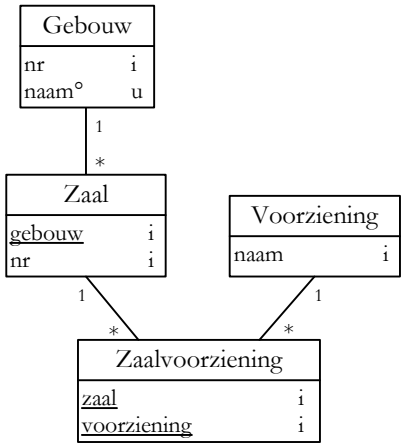
4.6 Atomaire informatiezinnen (met onderstreepte objectformuleringen):
- er is één album met albumnummer 10
- er is één album met albumnummer 24
- album 10 hoort tot genre 'pop'
- album 10 hoort tot genre 'jazz'
- album 24 hoort tot genre 'pop'

Indien er een betekenisvolle genrevolgorde is, komen hier nog de volgende informatiezinnen (feiten) bij:
- genre ‘pop’ bij album 10 heeft volgnummer 1
- genre ‘jazz’ bij album 10 heeft volgnummer 2
- genre ‘pop’ bij album 24 heeft volgnummer 1

4.7 De oplossing van figuur 4.14 staat toe om – zonder lege attribuutwaarden – per album precies het juiste aantal componisten op te geven, ook als dit aantal groter is dan drie en zonder lege ruimte te veroorzaken voor niet-bestaande componisten.
Een mogelijk nadeel is de wijze waarop componistinformatie in de applicatie wordt getoond: in een apart detailformulier. Dit kan echter ook als een voordeel worden gezien, vanwege het opsommende karakter.

4.8a Het diagram van figuur 4.19a is volledig afgestemd op de voorzieningen beamer, whiteboard en netwerkaansluiting. Het is daardoor niet flexibel. Het diagram van figuur 4.19b heeft een aparte tabel Voorziening, waarin de gebruiker voorzieningen naar keuze kan opnemen. Bijvoorbeeld ook een flip-over. Dit model biedt dus flexibiliteit. Dit leidt tot een voorlopige voorkeur voor het diagram van figuur 4.19b.

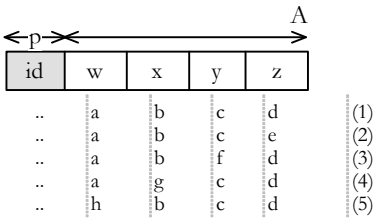
b Zie figuur 4.24.



Figuur 4.24 Transformatie van PIM naar PSM

c Het zaalformulier van de applicatie bij figuur 4.19a geeft de aanwezige voorzieningen weer via attributen, één attribuut per voorziening. In de applicatie bij figuur 4.19b moet je daarvoor een tabblad raadplegen. Wat het prettigste is laten we in het midden.

4.9 Figuur 4.25 geeft een minimaal illustratief informatievoorbeeld.



Figuur 4.25 Minimaal illustratief informatievoorbeeld bij identiteitsregel over vier attributen

Toelichting: de identiteitsregel houdt in dat geen strengere, 'smallere' regel geldt. Het informatievoorbeeld illustreert dit:
- rijen 1 en 2 illustreren dat geen identiteitsregel geldt over (w, x, y)
- rijen 1 en 3 illustreren dat geen identiteitsregel geldt over (w, x, z)
- rijen 1 en 4 illustreren dat geen identiteitsregel geldt over (w, y, z)
- rijen 1 en 5 illustreren dat geen identiteitsregel geldt over (x, y, z).

Nog strengere identiteitsregels (over twee kolommen of over één) gelden dus zeker niet.

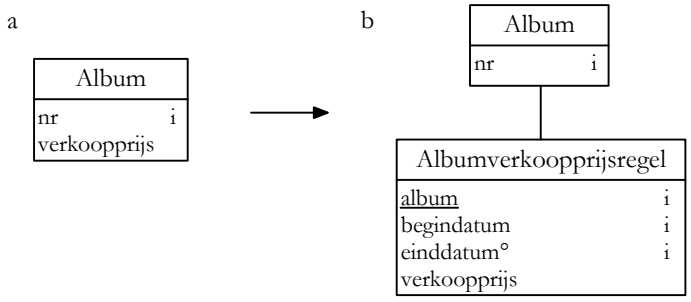
Uitwerkingen hoofdstuk 5

5.1 Wanneer elk verkocht albumexemplaar tot een aparte verkoopregel leidt, kan het aantal-attribuut worden weggelaten. Het aantal is dan immers altijd 1. Om dan toch mogelijk te maken dat van een album meerdere exemplaren worden verkocht binnen één transactie, moet de u-regel vervallen.

5.2a Verkoop.bedrag is afleidbaar als de som van de corresponderende Verkoopregel-bedragen.
Verkoopregel.bedrag is initieel afleidbaar als het product van aantal en van de waarde van verkoopprijs van het geassocieerde Album-object. In een gegeven Verkoop-object is de waarde van bedrag gelijk aan de som van de waarden van bedrag van de geassocieerde Verkoopregel-objecten.

b Om Verkoopregel.bedrag permanent afleidbaar te maken, moet niet alleen de huidige verkoopprijs gemodelleerd worden, maar ook de verkoopprijzen in het verleden. Anders gezegd: het model moet voorzien in het bewaren van de historie van verkoopprijzen.

Dit geeft een transformatie van 'één' naar 'veel', een type transformatie die vaak voorkomt. In het algemeen leidt dit tot het vervangen van een attribuut door een kindklasse. In dit geval moet het attribuut Album.verkoopprijs plaatsmaken voor een kindklasse met 'verkoopprijsregels'. Figuur 5.23 brengt de modeltransformatie in beeld.



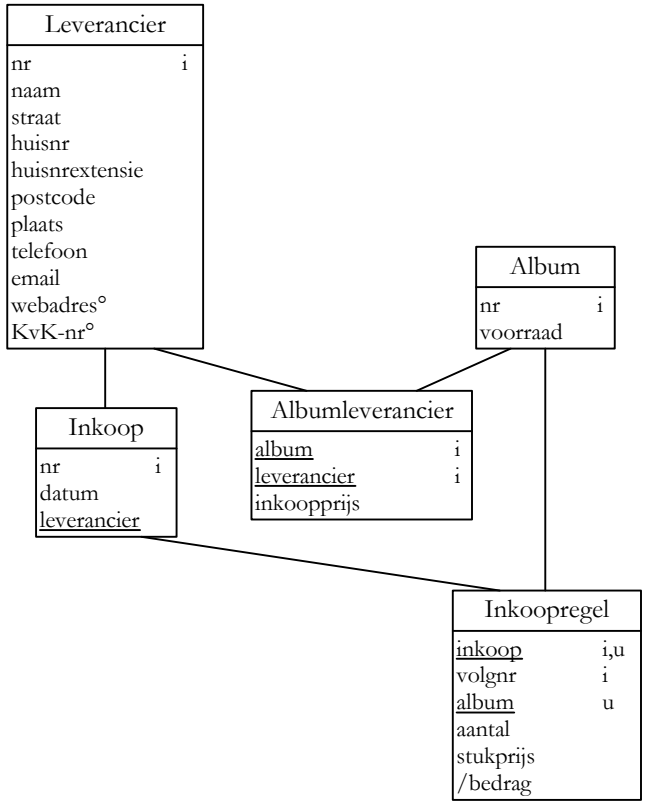
Figuur 5.23 Van één verkoopprijs naar een historie van verkoopprijzen

Elke verkoopprijsregel heeft behalve een bijbehorend album een datumperiode met de verkoopprijs die voor die periode geldig is. De meest recente datumperiode van een album heeft alleen een begindatum en bevat de 'huidige verkoopprijs'. Voor de perioden gelden regels analoog aan die van de kortingsintervallen van figuur 4.4b. De identificatieregel is net als die in figuur 4.4b 'breed' gekozen. Zie paragraaf 4.2.2 voor de overwegingen.

5.3a Album.inkoopprijs zal een update moeten ondergaan wanneer de standaardprijs door de betreffende leverancier wordt gewijzigd of wanneer een andere leverancier wordt gekozen.

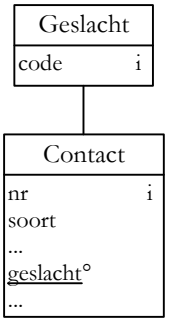
b Inkoopregel.stukprijs ondergaat in principe geen updates. Het is een transactiegegeven en daarmee een stukje historie. Historie verandert niet. U zou kunnen antwoorden: als er een fout is gemaakt.

c Zie figuur 5.24. Merk op dat de (standaard) inkoopprijs is verhuisd naar de koppelklasse AlbumLeverancier.



Figuur 5.24 Informatiediagram bij proces 'inkoop en voorraadbeheer' (deelproces inkoop)

5.4 Behalve een attribuut Contact.geslacht moet er nu een extra klasse komen om de mogelijke geslachten te standaardiseren. Zie figuur 5.25.



Figuur 5.25

De klasse Geslacht krijgt twee objecten: één met code 'm' en één met code 'v'. Een alternatieve oplossing is om de klasse Geslacht weg te laten en voor Contact.geslacht (dat dan een enkelvoudig attribuuttype krijgt) een *opsommingsregel* (value rule) te formuleren (met als lijst van toegestane waarden: 'm', 'v'). Value rules: zie bijlage van hoofdstuk 3.

5.5a De mogelijkheid om genres tevoren als stamgegevens in te voeren, zou vervallen. Genres zouden alleen nog – zonder opzoektabel en niet-gestandaardiseerd – kunnen worden ingevoerd als kenmerken van specifieke albums.

b Een genuanceerd antwoord is op zijn plaats: opzoeken is nog steeds mogelijk, maar doordat er geen standaardisatie is, kan het gemakkelijk gebeuren dat één genre bij verschillende albums onder verschillende omschrijvingen wordt geregistreerd, bijvoorbeeld: bebop naast Bebop of hiphop naast hip-hop.

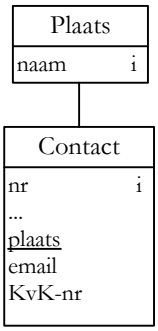
5.6a Een argument voor standaardisatie is dat het 'netjes' is om plaatsnamen correct te schrijven bij verzending van post. Een ander argument zou kunnen zijn dat bij zoekopdrachten alleen ingeval van standaardisatie zekerheid bestaat op een

volledig resultaat. Iets dergelijks geldt voor op plaatsnaam gesorteerde overzichten, natuurlijk alleen als aan zulke overzichten behoefte is.

Als argument tegen zou men kunnen aanvoeren dat tevoren alle plaatsnamen moeten worden ingevoerd, ook namen van plaatsen waar de muziekwinkel in de toekomst waarschijnlijk nooit een contact zal hebben. Hier zijn echter twee argumenten tegenin te brengen. Allereerst zijn lijsten met plaatsnamen gewoon beschikbaar en kan hieruit eenvoudig een SQL-insertscript worden gefabriceerd. Verder is het helemaal niet nodig alle plaatsnamen tevoren in te voeren, een plaatsnamentabel kan ook langzaam worden opgebouwd.

Een kleine, vooral lokaal opererende muziekwinkel met vaste leveranciers zal misschien geen behoefte hebben aan standaardisatie. Bij een grote muziekwinkel met contacten binnen een veel grotere straal zal die behoefte waarschijnlijk wel bestaan.

b Figuur 5.26 geeft een structuur waarbij plaatsnamen zijn gepromoveerd tot stamgegevens.

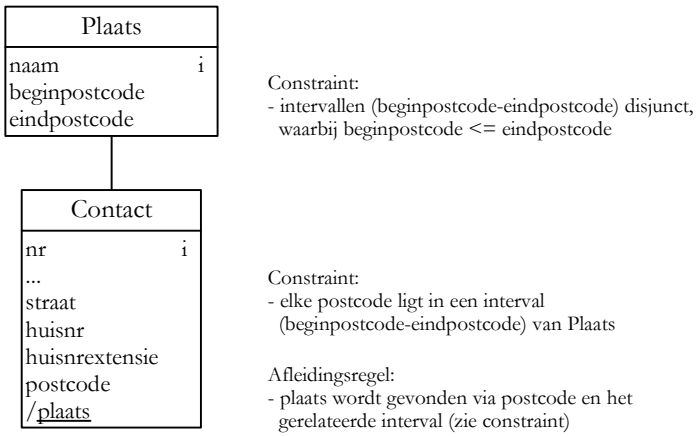


Figuur 5.26 Plaatsnamen gestandaardiseerd

Hierbij is uitgegaan van de plaatsnaam als identificerend kenmerk van een plaats. Deze aanname is gerechtvaardigd als we ons beperken tot gemeentenamen. De aanname is niet gerechtvaardigd als we alle plaatsnamen toelaten, zelfs niet als een provinciecode aan de plaatsnaam wordt toegevoegd. Zo zijn er bijvoorbeeld twee plaatsen Berg en Dal in Gelderland, binnen de gemeenten Groesbeek respectievelijk Ubbergen.

5.7 Zie figuur 5.27. Deze oplossing gaat uit van de volgende aannames.
- Plaatsnamen zijn uniek.
- Elke plaats heeft één uniek postcode-interval, waarbij de intervallen onderling disjunct zijn.

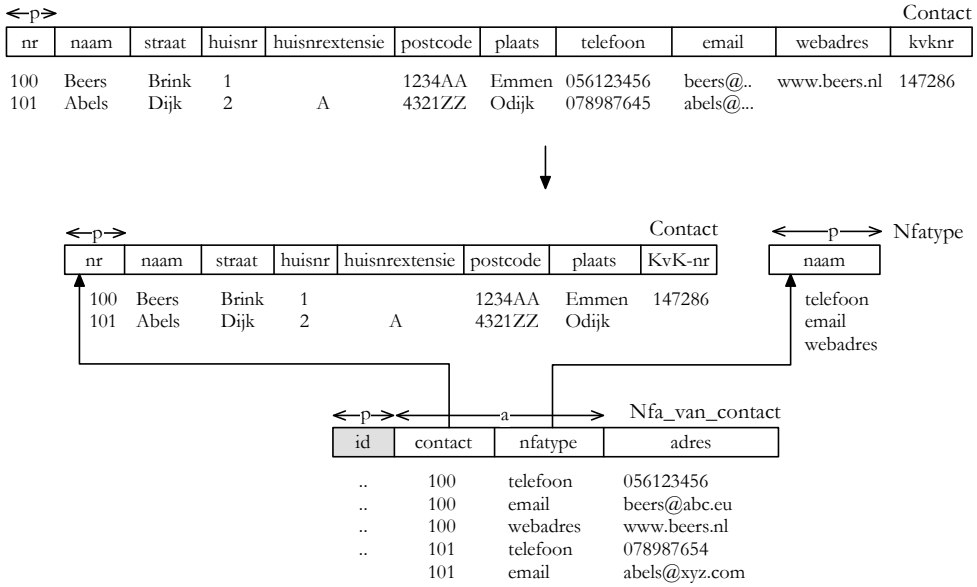
De plaatsnaam van een contact is nu afleidbaar door het juiste postcode-interval bij de postcode te zoeken. We hebben dit gemodelleerd via een afleidbaar associatieattribuut Contact.plaats.



Figuur 5.27 Plaatsnamen gestandaardiseerd en beschikbaar via de postcode (alleen voor gemeentenamen)

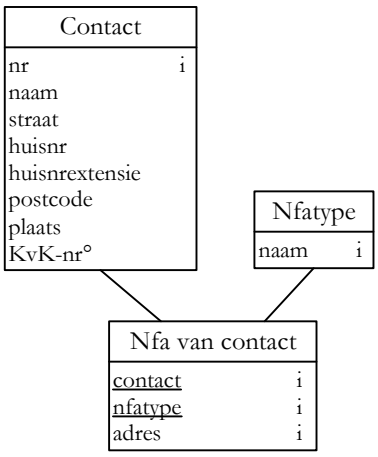
Opmerkingen
- Dit is niet het enige goede model, maar wel een eenvoudig en helder model.
- Postcodegegevens, gekoppeld aan namen van straten en plaatsen, zijn commercieel beschikbaar, bijvoorbeeld als spreadsheet of als tekstbestand. Conversie naar een specifiek databaseformaat is eenvoudig.

5.8a Zie figuur 5.28. NB de telefoonnummers in deze cursus, zowel in tekst als populaties, tellen met opzet slechts 9 cijfers.



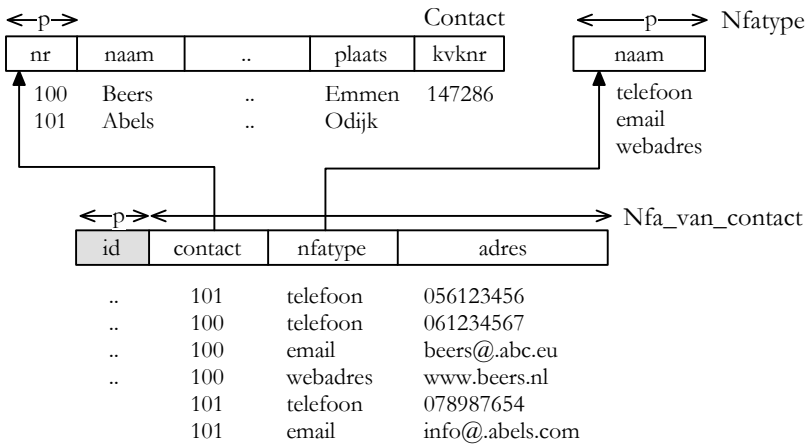
Figuur 5.28 PSM's met illustratieve populaties bij figuur 5.19

b De combinatie (contact, nfatype) in 'Nfa van contact' is nu niet langer uniek: de identiteitsregel moet worden verbreed met adres (zie figuur 5.29).



Figuur 5.29 Meerdere niet-fysieke adressen van één type toegestaan

Figuur 5.30 geeft hierbij een PSM met een illustratieve populatie. Merk op dat de alternatieve sleutel is verbreed en dat klant 101 twee telefoonnummers heeft.

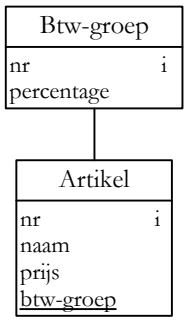


Figuur 5.30 PSM met illustratieve populatie bij figuur 5.30

5.9a Btw-percentages zijn stabieler dan prijzen en dus ook stabieler dan btw-bedragen. Het model waarbij 'btw' een bedrag aanduidt, is daarom minder aanvaardbaar dan het model waarbij 'btw' een percentage aanduidt.

b In de praktijk komt maar een beperkt aantal btw-percentages voor, die niet zijn gekoppeld aan afzonderlijke artikelen, maar aan van overheidswege bepaalde btw-groepen. Het diagram van figuur 5.22 (ongeacht de interpretatie van het attribuut btw) impliceert een update van wellicht vele artikelen zodra de overheid een btw-percentage verandert.

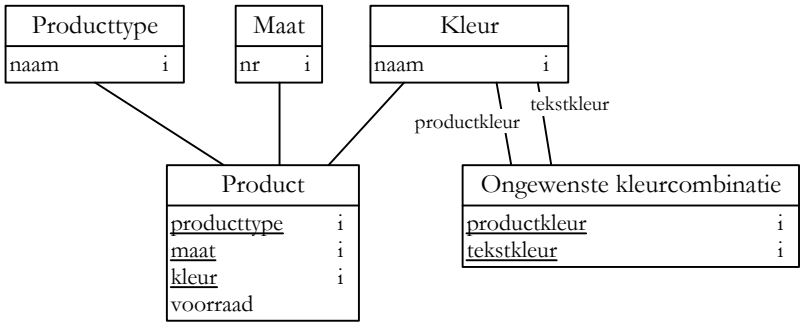
c Beter is de btw-percentages op te slaan volgens het principe 'single point of definition'. Hiertoe moet het begrip 'btw-groep' worden gemodelleerd, zie figuur 5.31. Aangezien de btw-groep van een artikel stabieler is dan het btw-percentage, is dit een hele verbetering.



Figuur 5.31 Modellering van btw, via btw-groepen

5.10a Het is verstandig het model op te zetten vanuit de meest eenvoudige typen van informatie: de gestandaardiseerde producten (T-shirts, poloshirts, ...), maten (S, XL, ...) en kleuren (lila, zeegroen, ...). Dit zijn typisch stamgegevens, die later in de applicatie in lookup-lijstjes beschikbaar moeten komen. We moeten nu goed nadenken over de terminologie, want het woord 'product' komt in de beschrijving voor als homoniem: we hebben 'producten' in algemene zin (T-shirts) maar ook meer specifieke 'producten' zoals de klant die kan bestellen (een zeegroen T-shirt, maat S). Om het onderscheid te maken zullen we het vanaf nu hebben over producttypen (T-shirt) en producten (zeegroen T-shirt, maat S).

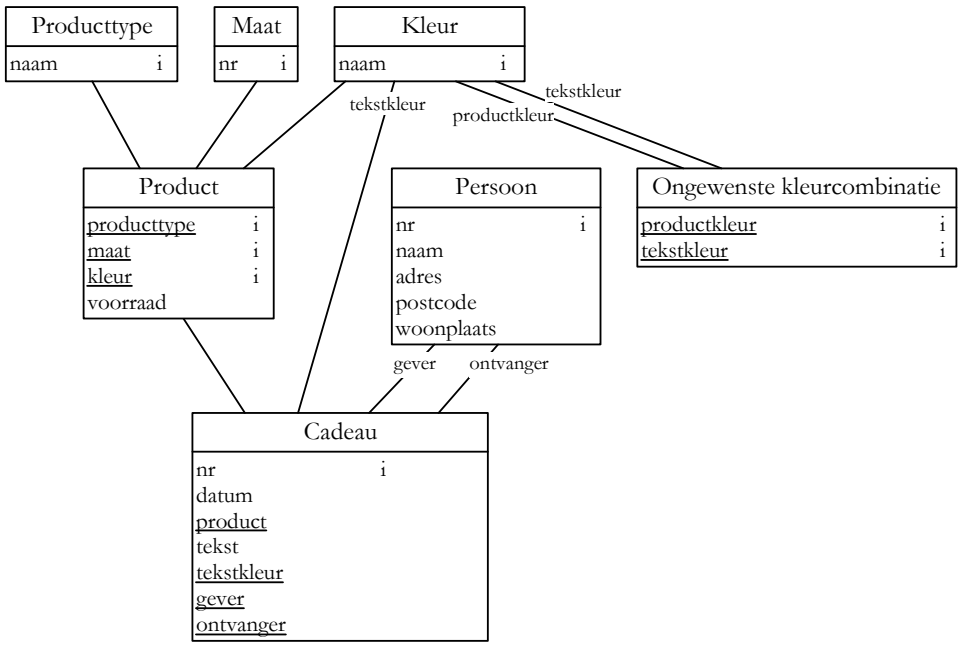
Dit leidt allereerst tot drie eenvoudige klassen: Producttype, Maat en Kleur. We kunnen verwachten dat deze drie klassen definitief 'bovenin' het model komen te staan. Zie figuur 5.32. De klasse Product, met een voorraad-attribuut, wordt hiervan een gemeenschappelijke kindklasse. Merk op dat Product een 3-voudige identificatieregel heeft. Tenslotte breiden we het aanbodmodel nog uit met een klasse voor de 'ongewenste kleurcombinaties'. Dit wordt een kindklasse van de klasse Kleur, met twee associaties naar Kleur.



Figuur 5.32 Informatiediagram cadeauservice (cadeauaanbod)

b Nu komen de echte cadeaus erbij: producten, geschonken door een gever aan een begunstigde. Laten we met de gever en de begunstigde beginnen: de gever is een echte klant van de cadeauservice. Beide echter zijn personen, waarvan dezelfde soort gegevens moeten worden bijgehouden. Het ligt daarom voor de hand om zowel gever als begunstigde te modelleren door een klasse Persoon. Omdat personen lastig te identificeren zijn zonder een kunstmatig attribuut, kiezen we voor een identificerend nummer. Zie figuur 5.33.

Dan de cadeaus. Deze leiden tot een klasse Cadeau, met associatieattributen voor de besteldatum, het bestelde product, de gever en de begunstigde. Ook moeten we denken aan de gewenste opdruk. Dit leidt tot nog twee attributen: tekst (voor de tekst van de opdruk) en tekstkleur. Omdat kleuren gestandaardiseerd zijn, wordt dit laatste een associatieattribuut met geassocieerde klasse Kleur.
Ter identificatie hebben we een kunstmatig nummerattribuut toegevoegd. Andere attributen en attribuutcombinaties zijn niet bruikbaar. (Het is zelfs zo dat dezelfde gever op dezelfde datum aan dezelfde begunstigde hetzelfde product met dezelfde tekst en dezelfde tekstkleur kan schenken!)



Figuur 5.33 Informatiemodel cadeauservice, uitgebreid voor transacties

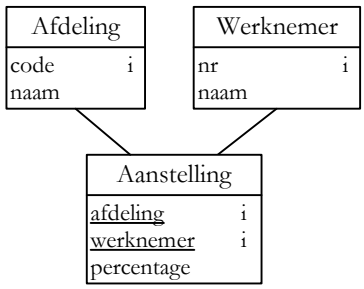
Via de associatie van Cadeau met Product kan worden achterhaald of het product in voorraad is. Zo niet, dan kan de applicatie dit melden aan de gever, zodat deze eventueel een andere keuze kan maken.

c We noemen twee mogelijke tekortkomingen van het model van figuur 5.33:
- Alle ooit als cadeau geleverde producten moeten in de populatie van Product aanwezig zijn, ook al zijn ze misschien uit het assortiment gehaald. Dit kan worden opgelost door aan Product een attribuut 'leverbaar' toe te voegen, waarvan de waarde ('ja' of 'nee') losstaat van de voorraad. De daarbij horende bedrijfsregel luidt dat alleen producten besteld kunnen worden waarvoor leverbaar = 'ja' (naast de regel die eist dat de voorraad groter dan 0 is).

– Gevers worden in dit model niet echt als klanten gemodelleerd. Het attribuut klant.nr is dan ook geen echt klantnummer, want ook begunstigden hebben zo'n nummer. Het is daardoor met dit model minder goed mogelijk (hoewel niet onmogelijk) om terugkerende klanten te herkennen. In hoofdstuk 6 komen we met aanpassingen van dit model, waarbij dit probleem is verholpen.

5.11a Relevante vragen zijn:
– Wat is een aanstelling precies?
– Is een aanstelling altijd bij een afdeling?
– Kan een werknemer per afdeling meer dan één aanstelling hebben?

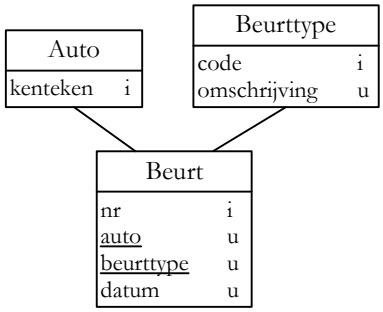
b De oplossing van figuur 5.34 gaat uit van de aanname dat een aanstelling feitelijk een koppeling betreft van een werknemer en een afdeling. Zo'n koppeling 'is' de aanstelling. Het percentage is in relatie tot een volledige weektaak.



Figuur 5.34 Aanname: een aanstelling 'is' de koppeling van een afdeling en een werknemer

Uitwerkingen hoofdstuk 6

6.1 Zie figuur 6.47 voor een informatiediagram.



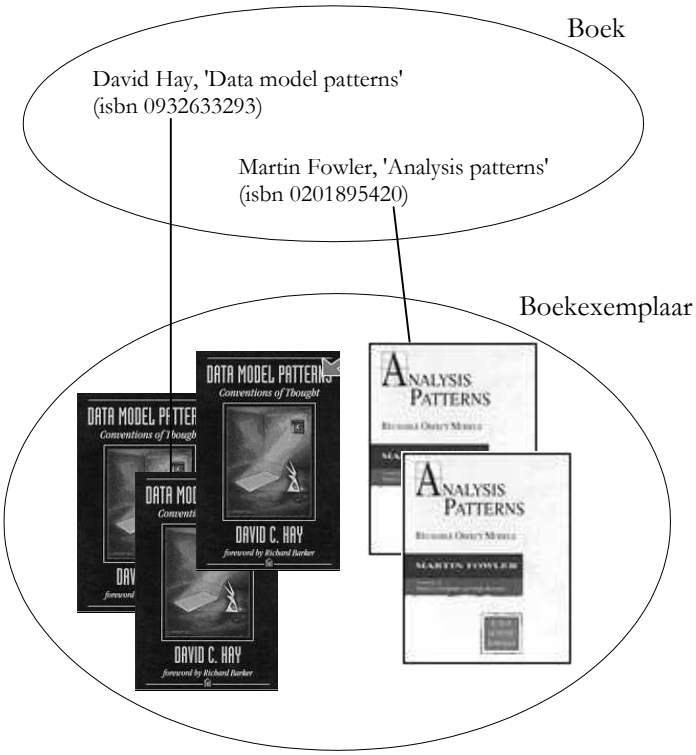
Figuur 6.47 Geen associatiepatroon!

Een beurt is niet te identificeren met de combinatie van een auto met een beurttype. Bij zo'n combinatie zijn meerdere beurten mogelijk, zoals we zien aan de uniciteitsregel die zich niet alleen uitstrekt over auto en beurttype maar ook over datum. Dit is dus geen voorbeeld van het associatiepatroon.

6.2 Gezien de attributen van C, E en F (waarvan niet bekend is of ze verplicht zijn) hoort bij een F-object maximaal één C-object en via dat object maximaal één A-object. Evenzo hoort er maximaal één B-object bij.
Bij een F-object kan dus een A,B-objectpaar horen, maar zeker niet meer dan één. De attribuutcombinatie a,b in D is echter niet uniek. Bij één A,B-objectpaar kunnen dus meer D-objecten horen.
Het antwoord luidt daarom: willekeurig veel. Figuur 6.12 is geen voorbeeld van het hartpatroon.

6.3 De uitspraak is niet correct. We illustreren dit aan de hand van figuur 6.48, waarin we naast het boek van Hay (met drie exemplaren) ook nog een boek van Fowler hebben opgenomen, met twee exemplaren.
We zien:
– Van de klasse Boek zijn twee instanties getekend.
– Van de klasse Boekexemplaar zijn vijf instanties getekend.
– De relatie tussen boekexemplaren en boeken is niet van het type 'instantie van'; het is een associatie: 'het boekexemplaar …hoort bij het boek …'.

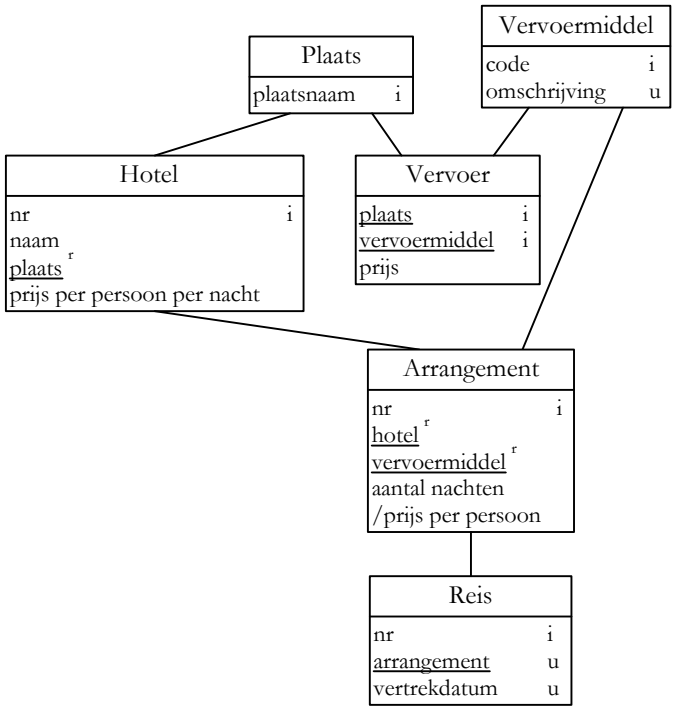
Algemeen: een ‘instantie’-relatie bestaat tussen een object en een klasse. Een object kun je zien als een element en de klasse als een verzameling. De relatie tussen objecten van verschillende klassen is van het type ‘associatie’.



Figuur 6.48 Boekexemplaren zijn geen instanties van een (abstract) boek

6.4a Wat het reisbureau aanbiedt en wat klanten boeken, zijn ‘concrete reizen’, dat wil zeggen: met een concrete vertrekdatum. Die concrete reizen zouden we ‘reisexemplaren’ kunnen noemen. Abstraheren we van de vertrekdatum, dan krijgen we ‘reizen’. In de tekst wordt maar op één plaats gerefereerd aan ‘reisexemplaren’: het woordje ‘reis’ in de laatste regel. Alle overige voorkomens van de woorden stedenreis, reis, reisarrangement en arrangement worden gebruikt voor het abstracte concept. Ze zijn synoniem. Homoniem zijn de voorkomens van ‘reis’ in de laatste twee regels.
Ook de beschrijving van de vervoermogelijkheden bevat nog synoniemen: vervoermiddel en vervoertype.

b Voor het model kiezen we in plaats van ‘reis’ en ‘reisexemplaar’ voor ‘arrangement’ (conform figuur 6.12) en ‘reis’. Zie figuur 6.49.
We kiezen conform figuur 6.12 voor ‘vervoertype’, waarbij ‘vervoer’ wordt gereserveerd voor een plaats-vervoermiddelcombinatie.

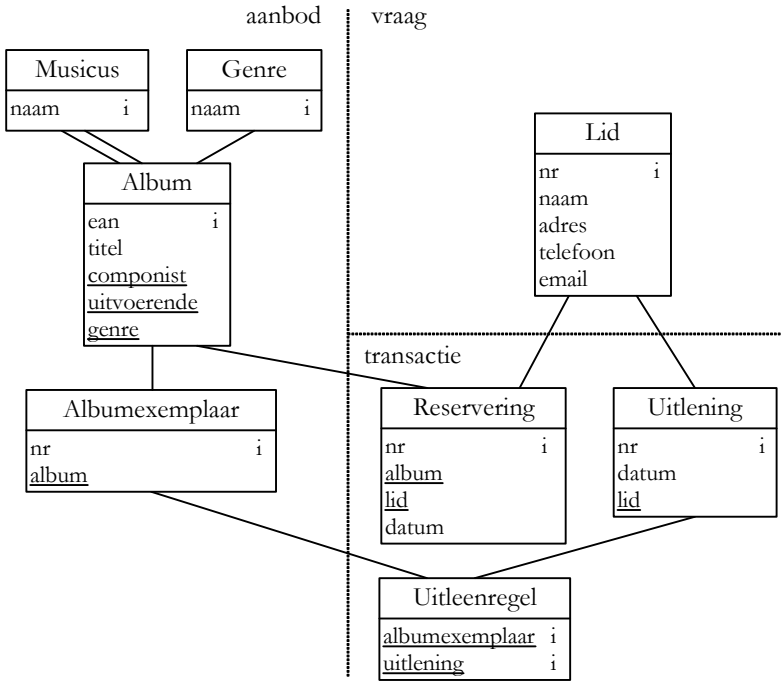


Figuur 6.49 Arrangementen en reizen: toepassing van exemplaarpatroon

Opmerking: de keuze voor het identificerende attribuut Reis.nr vloeit niet voort uit de tekst. Laten we dit weg, dan krijgt Reis een i-regel voor de attribuutcombinatie arrangement, vertrekdatum.

6.5a Zie figuur 6.50.

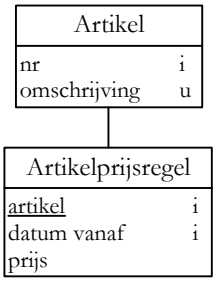
b Het Cathedronproject kent meerdere componisten, uitvoerenden en genres per album. Verder zijn ook uitleningen als enkelvoudige transacties gemodelleerd, net als reserveringen.



Figuur 6.50 Muziekuitleen met reserveringen als enkelvoudige transacties

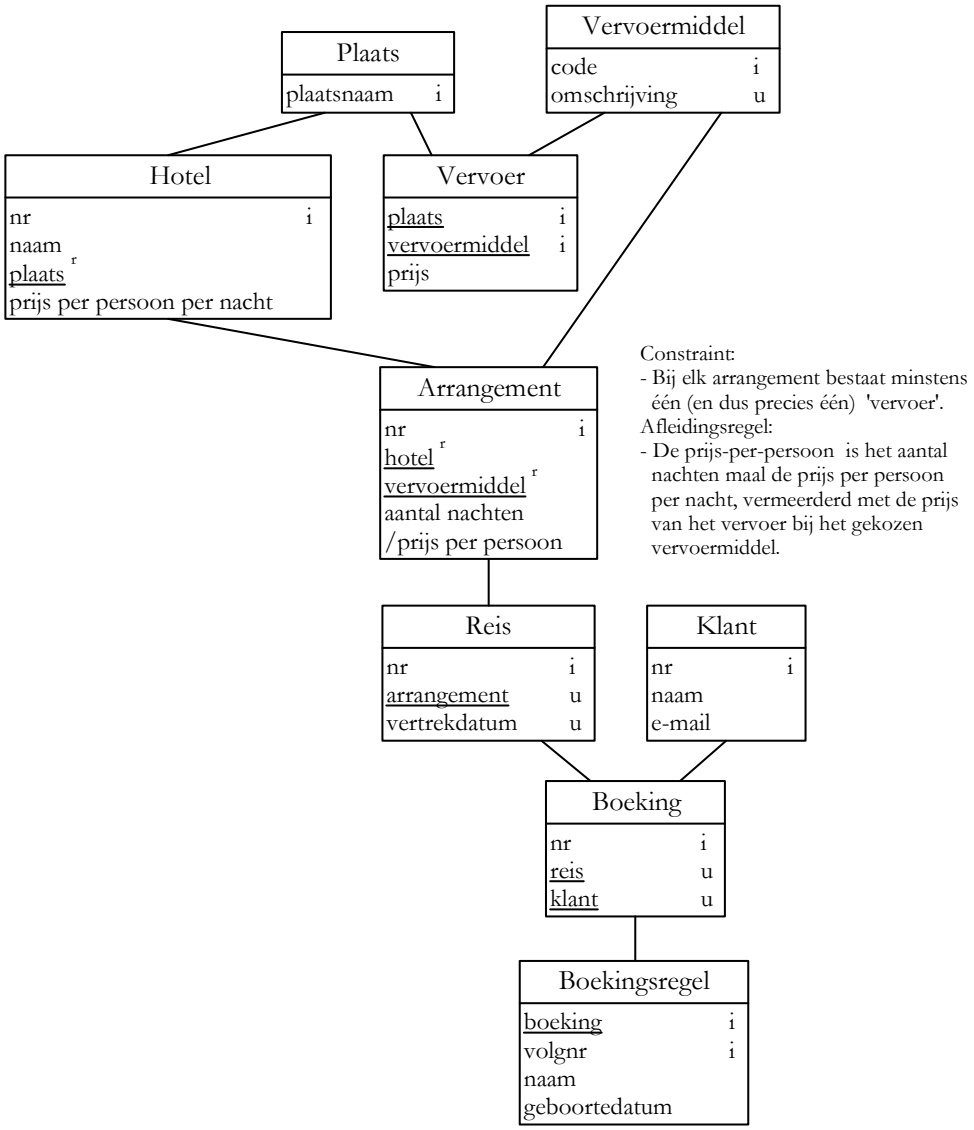
6.6a De uniciteitsregel houdt in dat per order een artikel maar op één orderregel mag voorkomen.

b Orderregel.bedrag is alleen afleidbaar als van artikelen de prijshistorie wordt bewaard. Hiervoor is een detailklasse vereist van Artikel, waarin de prijzen van artikelen worden vastgelegd in combinatie met een bepaalde periode of begindatum. Zie figuur 6.51. Merk op dat de naam van de prijshistorieklasse als altijd correspondeert met één instantie ervan. In plaats van alleen de 'datum vanaf' kan ook een periode worden gemodelleerd door een optionele 'datum tot en met' toe te voegen. Dit vraagt dan wel om een intervalconstraint (zie paragraaf 4.2.2).



Figuur 6.51 Ordermodel (fragment) met prijshistorie

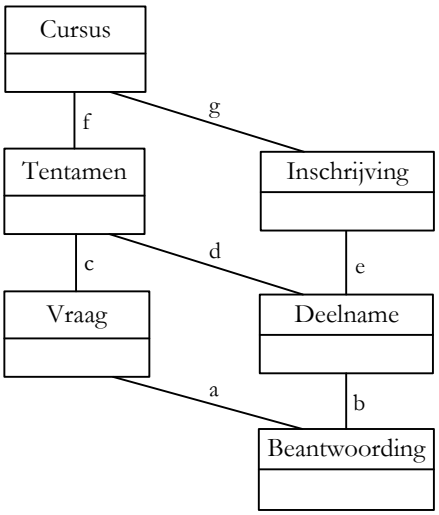
6.7 Figuur 6.52 toont een oplossing met samengestelde transacties. De (gebundelde) transacties zijn boekingen. De transactieregels heten nu boekingsregels en hebben betrekking op een reisgenoot. Reisgenoten zijn slechts bekend via geboortedatum en naam (voorletters en achternaam).
Merk op dat de structuur afwijkt van het standaard ordermodel. Een transactie heeft immers betrekking op maar één artikel (reis).



Figuur 6.52 Reisbureau, uitgebreid met boekingen

Opmerking: alleen al wettelijke regels en richtlijnen van de ANVR (Algemeen Nederlands Verbond van Reisondernemingen) zullen wellicht aanleiding zijn tot een flinke bijstelling van het model.

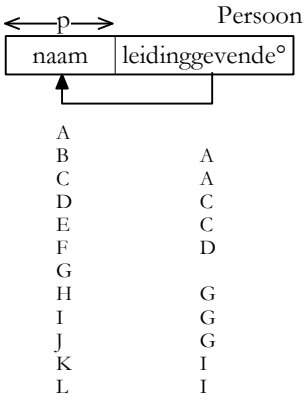
6.8 Zie figuur 6.53.



Figuur 6.53 Drie paden van Beantwoording naar Cursus

Ga uit van een Beantwoording en het pad acf. Stel, u vindt via dit pad cursus c. Diezelfde cursus c zou u ook gevonden hebben via het pad bdf, omdat de deelpaden ac en bd tot hetzelfde tentamen leiden. In dit nieuwe pad bdf kan het deelpad df worden vervangen door eg, waarbij u nog steeds bij dezelfde cursus uitkomt. Dit geeft het pad beg, zoals gevraagd.

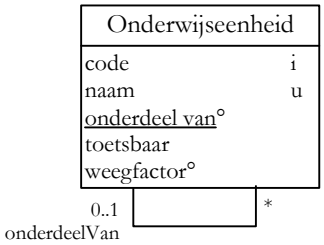
6.9 Zie figuur 6.54.



Figuur 6.54 PSM bij hiërarchie van figuur 6.28

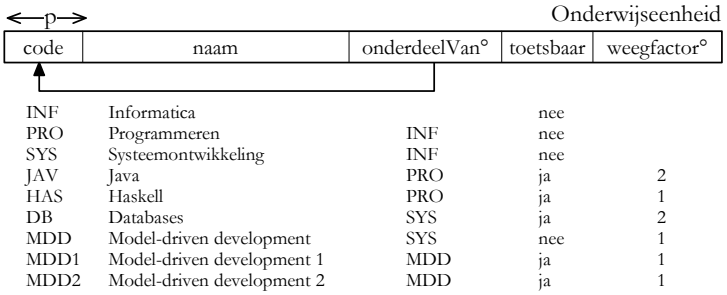
6.10 De keuze om de studierichting ook als onderwijseenheid te zien leidt tot één klasse Onderwijseenheid met unieke code en naam. De code kiezen we ter identificatie (i), de naam krijgt een gewone uniciteitsregel (u). Alle onderwijseenheden in figuur 6.33b behalve Informatica (de root) hebben een ouderelement. Dit geeft een optioneel attribuut, dat we 'onderdeel van' noemen.

De beschrijving impliceert dat vakken wel of niet toetsbaar zijn. Dit modelleren we via een boolean attribuut toetsbaar. Tot slot moeten we de weegfactoren modelleren. Op het eerste gezicht lijken deze aan twee studieonderdelen te 'hangen'. Bijvoorbeeld: 'Java heeft weegfactor 2 ten opzichte van Programmeren'. Omdat elk kind echter maar één ouder heeft, is de weegfactor een eigenschap van alleen het kind: 'Java heeft weegfactor 2'. Dus wordt weegfactor een (optioneel) attribuut van Onderwijseenheid. Het resultaat zien we in figuur 6.55.



Figuur 6.55 Informatiediagram hiërarchisch geordend onderwijsprogramma

b Zie figuur 6.56.

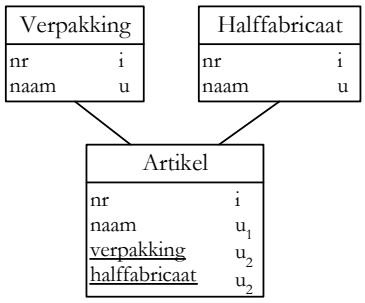


Figuur 6.56 PSM bij onderwijsprogrammahiërarchie van figuur 6.33

We merken nog op dat weegfactoren op de juiste plaatsen moeten zijn ingevuld om later (wanneer het model is uitgebreid met studentgegevens) tentamencijfers correct van beneden naar boven door te kunnen rekenen. Hiertoe moeten nog bedrijfsregels aan het model worden toegevoegd, eventueel in combinatie met workflowachtige regels van het type 'uiteindelijk verplicht' (zie paragraaf 5.3.7).

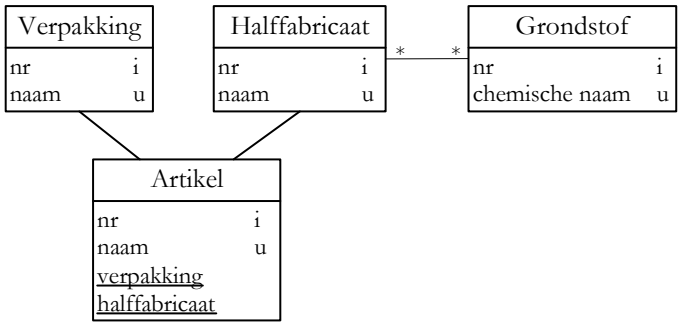
6.11 Stel dat één Cadeau-object twee verschillende Begunstigde-objecten als kind had. Die twee kindobjecten zouden dan dezelfde ouder hebben, dus dezelfde waarde van Begunstigde.cadeau. Dit is in strijd met de identificatieregel.

6.12 Uit de beschrijving halen we allereerst dat een artikel een uniek artikelnummer heeft. Dit leidt tot een i-regel voor artikelnummer. Ook de artikelnaam is uniek (eerste uniciteitsregel: u_1).
Vervolgens blijkt uit de beschrijving dat een artikel beschouwd kan worden als een halffabricaat in verpakking. Hierin herkennen we het associatiepatroon (zie figuur 6.57). Artikel is associatieklasse van Halffabricaat en Verpakking (een tweede uniciteitsregel: u_2).



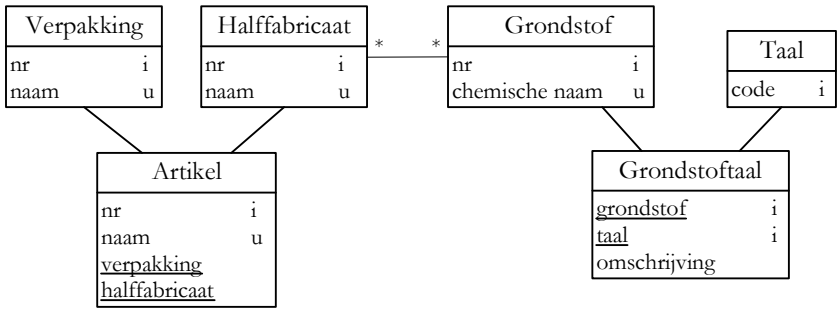
Figuur 6.57 Artikel is halffabricaat in verpakking

Volgens de beschrijving is een halffabricaat gemaakt uit verschillende grondstoffen en kan een grondstof in verschillende halffabricaten voorkomen. Dit geeft een n-op-m-associatie, zie figuur 6.58. Zolang voor halffabricaat-grondstof-combinaties geen eigen attributen hoeven te worden weergegeven, voldoet dit plaatje uitstekend als communicatiemiddel. Desgewenst kan de n-op-m-associatie worden vervangen door een associatieklasse.



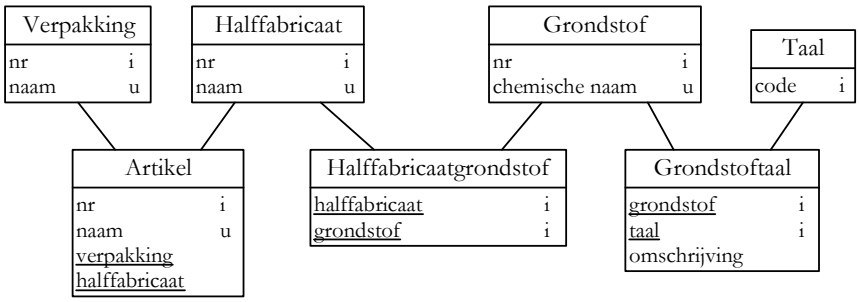
Figuur 6.58 Halffabricaat-Grondstof: n-op-m

Vervolgens hebben we te maken met grondstofomschrijvingen in verschillende talen. Het ligt voor de hand dat de talen zijn gestandaardiseerd. We nemen aan dat dat gebeurt via een code voor elke taal (bijvoorbeeld ENG voor Engels). Zie figuur 6.59.



Figuur 6.59 Grondstofomschrijvingen zijn er in meerdere talen

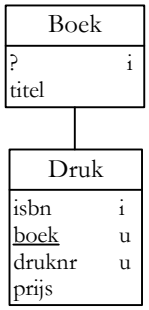
Desgewenst kunnen we nog de n-op-m-associatie van figuur 6.59 weergeven via een associatieklasse, zie figuur 6.60. Merk op dat het associatiepatroon drie keer voorkomt, in de vorm van een keten.



Figuur 6.60 Alle n-op-m-associaties uitgemodelleerd tot associatieklassen

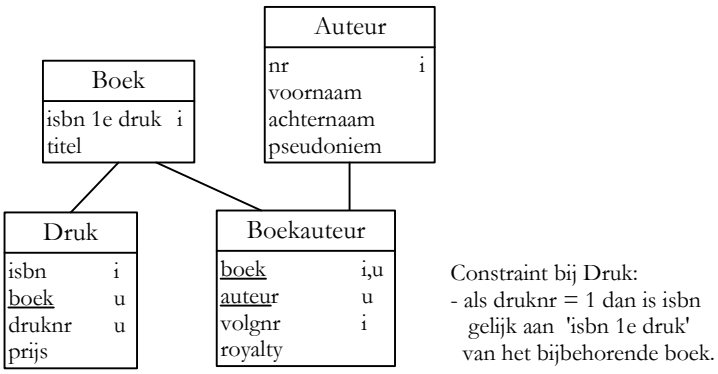
6.13a Alleen de titel is drukonafhankelijk; elke druk heeft een eigen isbn en de prijs kán (en zal meestal) per druk verschillen.

b Als boeken (in algemene, drukonafhankelijke zin) en boekdrukken zich verhouden zoals in het exemplaarpatroon, dan leidt dit tot de structuur van figuur 6.61. Het probleem is al gemarkeerd: hoe moeten we Boek identificeren? De oorspronkelijke identificatie van Boek is immers naar Druk verhuisd.



Figuur 6.61 Boeken en drukken: exemplaarpatroon?

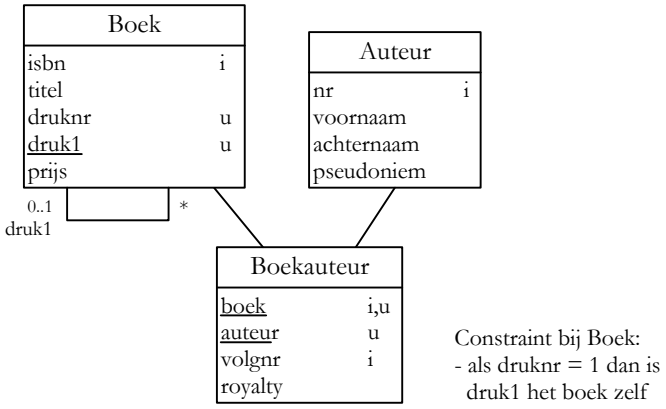
Is wellicht een deel van het isbn drukonafhankelijk, zodat we dat als identificatie van Boek kunnen gebruiken? Dit blijkt niet zo te zijn. Moeten we dan een nieuwe, drukonafhankelijke identificatie invoeren? In figuur 6.62 hebben we voor een andere oplossing gekozen: als identificatie van Boek kiezen we het isbn van de eerste druk (isbn 1e druk).



Figuur 6.62 Uitwerking van exemplaarpatroon

We hebben aangenomen dat de auteurs en ook het royaltypercentage drukonafhankelijk zijn. Het is zeer de vraag of dit correct is!

c Zie figuur 6.63 voor een oplossing waarbij Boek en Druk van figuur 6.61 in elkaar zijn geschoven. Het idee om het drukonafhankelijke 'boek' te identificeren door het isbn van de eerste druk, is gehandhaafd. De n-op-1-associatie van Druk met Boek is overgegaan in een 1-op-n-associatie van Boek met zichzelf, een recursieve associatie dus.



Figuur 6.63 Uitwerking van exemplaarpatroon

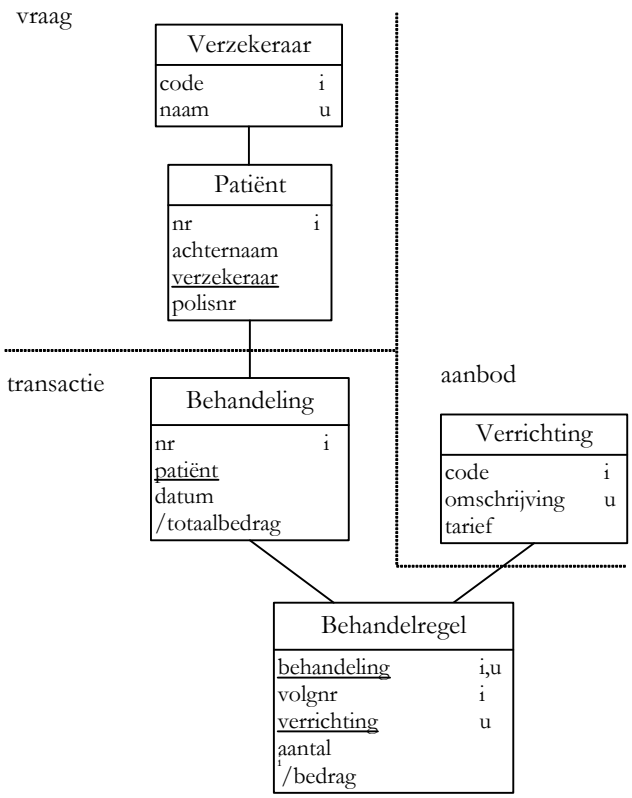
Men kan nog twisten of de 0..1-multipliciteit niet een 1-multipliciteit moet zijn, waarbij dus Boek.druk1 verplicht is. In dat geval wordt het attribuut Boek.druk1 ook ingevuld voor de eerste druk zelf.

d De auteurs en ook het royaltypercentage kunnen in figuur 6.62 per druk verschillen. Zelfs de titel kan anders zijn. Desgewenst kan als constraint worden toegevoegd dat gelijke drukken gelijke titels hebben. Of de titel wordt alleen bij

druk 1 vermeld, waarbij wordt afgesproken dat deze voor alle drukken geldt. (De applicatie moet die titel dan bij alle drukken tonen, uiteraard.)
De tweede oplossing is het eenvoudigst en het meest flexibel en heeft daarom onze voorkeur.

6.14 Dit is een typisch vraag- en aanbodprobleem. Door de aanbod-, vraag- en transactiekant los van elkaar en in de juiste volgorde te modelleren, valt het probleem uiteen in drie deelproblemen van geringe complexiteit.

Aanbodkant
De door de tandarts aangeboden 'artikelen' zijn de gestandaardiseerde verrichtingen. Deze bestaan onafhankelijk van patiënten die deze verrichtingen tijdens een behandeling ondergaan. Dit geeft aanleiding tot een klasse Verrichting (zie figuur 6.64).

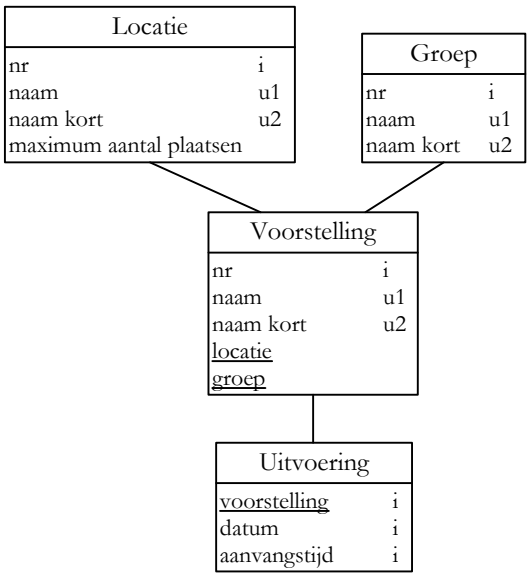


Figuur 6.64 Typisch vraag- en aanbodmodel voor tandartspraktijk

Vraagkant
De vraagkant wordt primair gevormd door de 'klanten', die bij de tandarts patiënt heten. De patiënten hebben we in figuur 6.64 gemodelleerd als een klasse Patiënt, met een kunstmatig patiëntnummer en twee attributen voor verzekeringsgegevens: een voor een verwijzing naar de verzekeraar (associatieattribuut, wijzend naar klasse Verzekeraar) en een voor het polisnummer. We hebben aangenomen dat verzekeraars naast een naam een gestandaardiseerde code hebben.
Overige voor de hand liggende Klant-attributen (voorletters, naam, geboortedatum, geslacht, fysieke en niet-fysieke adresgegevens) zijn weggelaten.

Transactiekant
Een tandheelkundige transactie heet een behandeling. Zoals een order orderregels heeft, zo heeft een behandeling behandelregels. Elke behandelregel specificeert een verrichting die in 'aantal'-voud is toegepast. Behandelregel.bedrag is initieel afleidbaar als het product van tarief en aantal.
De omschrijving sluit niet uit dat een patiënt meerdere behandelingen op één datum ondergaat. (Welke constraint komt er bij wanneer moet worden afgedwongen dat alle verrichtingen bij een patiënt op één dag samen één behandeling vormen?)

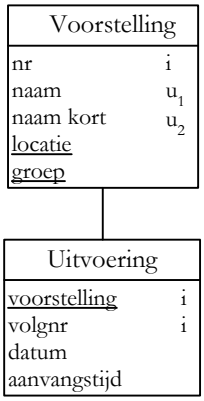
6.15a Zie figuur 6.65 (klassen Locatie, Groep en Voorstelling). Merk op dat de naam- en naamKort-attributen elk apart uniek zijn en daarom een genummerde uniciteitsregel hebben.



Figuur 6.65 Informatiediagram voorstellingsaanbod theaterfestival

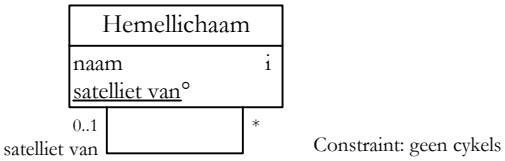
b In de beschrijving is sprake van voorstellingen en uitvoeringen. Een uitvoering is te beschouwen als een exemplaar van een voorstelling. Een 'voorstelling' is een relatief abstract, niet-tijdgebonden begrip en een 'uitvoering' is een concreet evenement waarvoor men een plaatsbewijs kan kopen. Zie figuur 6.65, waarin Voorstelling en Uitvoering samen een voorkomen van het exemplaarpatroon vormen. Ga na dat voor Uitvoering de drievoudige identiteitsregels geldt, zoals aangegeven.

Opmerking: identiteitsregels moeten zo stabiel mogelijk zijn. De identiteitsregel voor Uitvoering is daarom voor kritiek vatbaar, aanvangstijden willen immers nog wel eens veranderen. Te overwegen is daarom de uitvoeringen van een concert van elkaar te onderscheiden via een betekenisloos volgnummer, zoals in het exemplaarpatroon van figuur 6.16. Dit is een stabiele keuze, die ons vrijlaat om beperkingen te verbinden aan de datum- en tijdattributen (en deze later misschien weer te veranderen), zonder dat dit invloed heeft op de structuur. Zie figuur 6.66.



Figuur 6.66 Klasse Uitvoering met stabiele identiteitsregel

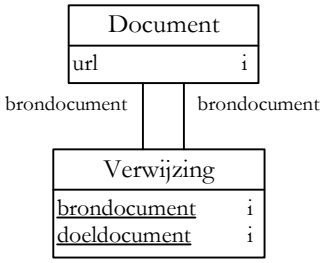
6.16 Het bijschrift bij figuur 6.46 suggereert dat we er misschien goed aan doen om sterren, planeten en manen te generaliseren tot één type van objecten: hemellichamen. Het informatiemodel moet dan uitdrukken dat het ene hemellichaam een satelliet is van het andere, via een strikte hiërarchie (zie figuur 6.67).



Figuur 6.67 Informatiemodel sterren, planeten en manen

De 'geen cykels'-regel impliceert dat een hemellichaam geen satelliet kan zijn van zichzelf, ook niet indirect via andere hemellichamen.

6.17 De structuur is die van een netwerk: een document kan naar meerdere (doel)documenten verwijzen en zelf het doel zijn van verwijzingen vanuit meerdere (bron)documenten. Zie figuur 6.68.



Figuur 6.68 Informatiediagram webhyperlinks

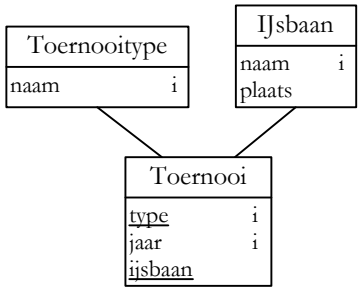
Uitwerkingen hoofdstuk 7

7.1a Stel, twee rijders, A en B, rijden tegen elkaar. Rijder A maakt een foute wissel waardoor rijder B wordt benadeeld. Rijder B zal dan gelegenheid krijgen de rit over te doen.

b Een probleem dat zich kan voordoen is dat er meer dan één kandidaat is voor een medaille, door gelijke eindtijden. Denkbare oplossingen zijn loting of het uitreiken van bijvoorbeeld twee gouden medailles en een bronzen medaille maar geen zilveren. Zonder loting kan het theoretisch voorkomen dat drie en zelfs nog meer gouden medailles moeten worden uitgereikt.
Hoewel de kans op dit soort 'incidenten' klein is (door tijdmeting tot in honderdsten van seconden), is hier uiteraard in voorzien in reglementen.

7.2 Bij wereldrecords, Europese records en dergelijke, en bij baanrecords kunnen we ons afvragen of een record op naam kan staan van meer rijders die een gelijke recordtijd hebben gereden. Als dat zo is en we dat ook moeten registreren, heeft dat gevolgen voor het model.
We hebben hiervan geen voorbeelden kunnen vinden en gaan er in het vervolg van uit dat een nieuwe recordhouder de tijd van de vorige verbeterd moet hebben, zodat er altijd maar één recordhouder is per afstand. Hoewel … stel dat twee rijders of rijdsters in dezelfde rit een record in dezelfde tijd verbeteren. Wat dan? We vragen ons ernstig af of de reglementen hierin voorzien en laten deze puzzel voor wat hij is. En we houden het bij één recordhouder per afstand.
Ziet u overigens hoe we ons als ontwikkelaar in rap tempo ontwikkelen tot domeindeskundige? We denken zelfs aan dingen waar de reglementen wellicht niet eens in voorzien.

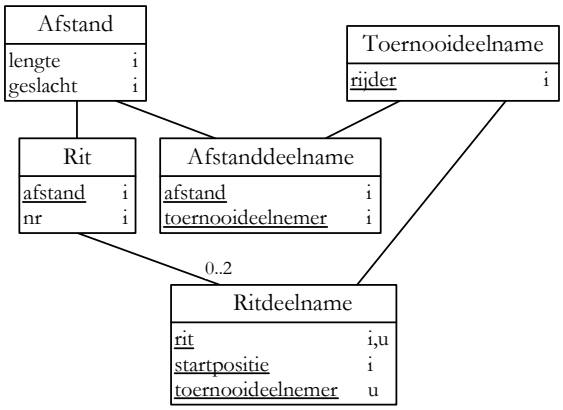
7.3 Bij een systeem voor meerdere toernooien worden ook klassen IJsbaan en Toernooi gemodelleerd. Dit zijn dan immers niet langer singletons. We krijgen dan de klassen van figuur 7.7 (nu met i-regels en niet als singleton), aangevuld met een klasse Toernooitype (zie figuur 7.23). De aanname is dat een toernooi van één toernooitype maar één keer per jaar kan plaatsvinden. Voor toernooien als WK allround is dit het geval, toch is dit iets waar nog goed over nagedacht moet worden.



Figuur 7.23 Informatiediagram aanbodkant voor systeem met meerdere toernooien

Opmerking: baanrecords moeten nu anders gemodelleerd worden.

7.4 Zie figuur 7.24 voor de vijf klassen in hun onderlinge samenhang. Puur visueel doet dit al denken aan het hartpatroon. Maar dat is niet genoeg. We moeten aantonen dat bij één instantie van Ritdeelnemer maximaal één instantie van Afstanddeelnemer hoort.



Figuur 7.24 Bij één ritdeelname hoort één afstanddeelname: hartpatroon

Welnu: bij een ritdeelname hoort, via de klasse Rit, één afstand. Ook hoort er één toernooideelname bij. Er hoort dus één combinatie van een afstand en een toernooideelname bij. Dat is precies de combinatie waardoor de klasse Afstanddeelname wordt geïdentificeerd. Bij één ritdeelname hoort dus maximaal één afstanddeelname. Via een aanvullende constraint kunnen we eisen dat dit 'precies één' is. Zie ook opgave 6.2.

7.5 Nee, wanneer we naar het tijdsverloop kijken, zien we dat eerst de afstanddeelnamen worden aangemaakt (bij de inschrijving voor een afstand). Pas later komen daar de ritten en ritdeelnamen bij. Daarbij geldt als constraint dat een rijder alleen aan een rit mag deelnemen wanneer die is ingeschreven als deelnemer aan die afstand.
Aanvankelijk is er dus geen redundantie en wanneer die er later wel is, is die tijdens het voorafgaande proces gecontroleerd door de constraint.

7.6a De vraag is of een rijder vaker dan één keer een rit kan rijden bij dezelfde afstand. In de realiteit kan dat inderdaad gebeuren, bijvoorbeeld wanneer de rijder door een fout van de tegenstander benadeeld is en de rit mag overdoen. Wil de schaatsliefhebber dergelijke finesses in het systeem vastleggen, dan zijn alleen de oplossingen 1 en 2 correct.

Wanneer de schaatsliefhebber alleen geldige ritten wil vastleggen die meetellen voor het eindklassement, dan is klasse Afstanddeelnemer voldoende om alle relevante wedstrijdinformatie vast te leggen. De oplossingen 3 en 4 komen dan in aanmerking.

b Moeten we kiezen uit de oplossingen 1 en 2, dan heeft oplossing 1 onze voorkeur, omdat de structuur minder 'diep' is. Voor het maken van formulieren is dat een voordeel. Zelfs de default-applicatie bij figuur 7.14 is al simpeler dan die bij figuur 7.15.
Uitgaande van een beperkte informatiebehoefte, komen we uit op de oplossingen 3 en 4. Oplossing 4 lijkt, als eenvoudigste, erg aantrekkelijk. Alle relevante wedstrijddata zijn op basis hiervan te onderhouden, zolang er maar niet meer dan één rit is per afstanddeelnemer.

Toch mist er iets: in de relevante wereld is 'rit' een belangrijk concept, maar er is geen klasse Rit. Ritten zijn gereduceerd tot een ritnummer binnen klasse Afstanddeelnemer. We merken dat wanneer we echte ritformulieren willen maken, bijvoorbeeld een overzicht van alle ritten per afstand met wie tegen wie rijdt. Voor zo'n formulier krijgen we niets kado. Bij oplossing 3 krijgen we tenminste al een default Rit-formulier kado, met details van bijbehorende afstanddeelnemers.

Al met al vinden we oplossing 4 – om in schaatstermen te spreken – wat al te kort door de bocht. Net een beetje té slim. Oplossing 3 vinden we als conceptueel model completer en dat beloont zich in de applicatie. De extra constraint nemen we daarbij voor lief.

7.7 Een persoonlijk record voor het klassementstotaal is een enkelvoudig kenmerk van een rijder. Hiervoor is dus een attribuut Rijder.klassementsrecord nodig.

Uitwerkingen hoofdstuk 8

8.1 De volgende SQL-query is equivalent met die van tabel 8.1:

```
select G.naam, (select count(Ag.genre)
                from   Albumgenre Ag
                where  Ag.genre = G.naam) aantal_albums
from   Genre G
```

8.2 Zie tabel 8.8. De masterdataset bevat een statistisch gegeven, waarvoor een join nodig is. De centrale tabellen, tevens starttabellen van de joinnavigatiepaden, zijn vetgedrukt.

Tabel 8.8 Datasets voor master-detailformulier Leveranciers

Formulier: Leveranciers	
Masterdataset	Leveranciers
Centrale tabel:	**Contact**

SQL-query:	`select C.nr, C.naam, C.straat, C.huisnr, C.huisnrextensie, C.postcode, C.plaats, C.KvK_nr, count(C.nr) aantal_albums`
	`from Contact C`
	`join Album A on C.nr = A.leverancier`
	`where C.soort = 'Leverancier'`
	`group by C.nr, C.naam, C.straat, C.huisnr, C.huisnrextensie, C.postcode, C.plaats, C.KvK_nr`

Detaildataset	Album
Centrale tabel:	**Album**
SQL-query:	`select A.nr, A.titel, A.inkoopprijs, A.verkoopprijs, A.voorraad, A.leverancier` `from Album A`
Linkfield:	leverancier

Toelichting : Voor degenen wier SQL-kennis wat opfrissing kan gebruiken: merk op dat we 'verfijnd groeperen' over de hele kolommenlijst van C.nr t/m C.KvK_nr. Conceptueel gezien zou groeperen over de primaire sleutel C.nr voldoende moeten zijn.

Analoog aan de uitwerking van opgave 8.2 geven we het volgende alternatief voor de SQL-query, zonder group by:

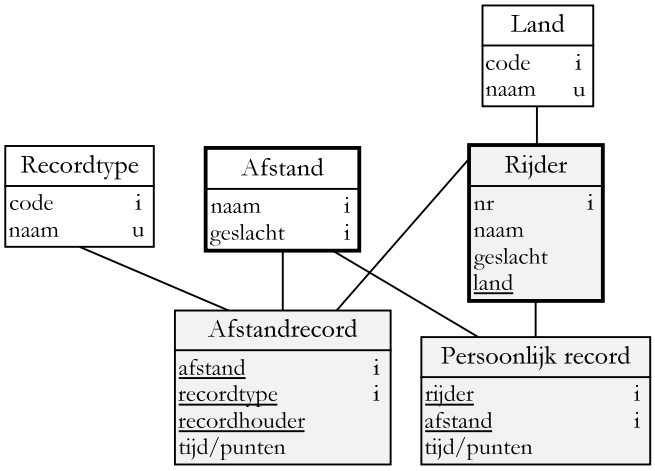
```
select C.nr, C.naam, C.straat, C.huisnr, C.huisnrextensie, C.postcode,
       C.plaats, C.KvK_nr, (select count(A.leverancier)
                            from   Album A
                            where  A.leverancier = C.nr) aantal_albums
from   Contact C
where  C.soort = 'Leverancier'
```

8.3 Zie tabel 8.9. De twee master-detailformulieren corresponderen met de gestippelde rechthoeken in het informatiediagram.

Tabel 8.9 Formulierspecificatie 'rijders en records'

Proces 'rijders en records'

Informatiediagram



formulieren

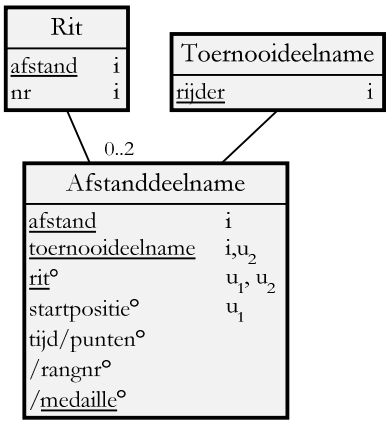
naam	*beknopte beschrijving*	*centrale klasse*
Rijders	beheerformulier voor rijders detailframe voor persoonlijke records	Rijder
Records per afstand	beheerformulier voor afstanden detailframe voor afstandrecords	Afstand (R)

8.4 Zie tabel 8.8.

Tabel 8.10 Formulierspecificatie toernooi

Proces toernooi

Informatiediagram



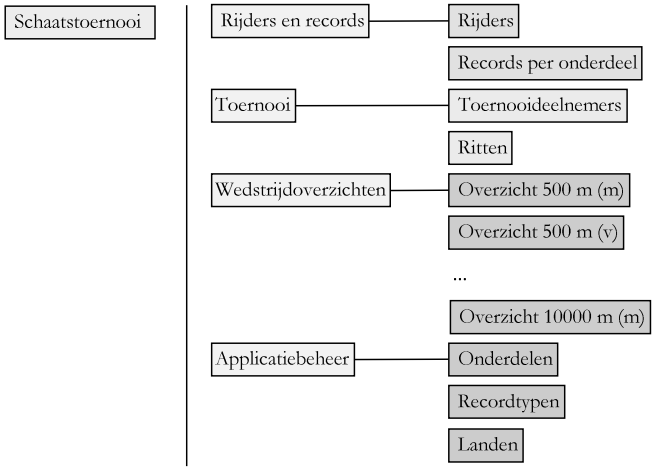
formulieren

naam	*beknopte beschrijving*	*centrale klasse*
Toernooideelnemers	– overzicht toernooideelnemers – detailframe voor gegevens verreden afstanden	Toernooideelname
Ritten	– overzicht ritten – detailform voor ritdeelnemers	Rit
Overzicht 500 m (m)	– wedstrijdoverzicht 500 m mannen	Afstanddeelname (R)
Overzicht 500 m (v)	– wedstrijdoverzicht 500 m vrouwen	Afstanddeelname (R)
etc.		...
Totaaloverzicht	– totaaloverzicht met klassement	Afstanddeelname (R)

Opmerking: formulier Toernooideelname bevat alle wedstrijdresultaten per toernooideelnemer. Deeloverzichten, bijvoorbeeld over één bepaalde afstand of van alleen het klassement, kunnen behalve door specifieke formulieren (met alleen R-functionaliteit) worden verkregen door zoekopdrachten (filteropdrachten).

Raadpleeg het project Schaatstoernooi2 om te zien hoe we de beknopte specificaties hebben uitgewerkt tot bruikbare formulieren.

8.5 Een menustructuur is eenvoudig te baseren op de drie informatiebeheerprocessen. Daarnaast voegen we een optie 'wedstrijdoverzichten' toe. Zie figuur 8.13. De optie 'wedstrijdoverzichten' is te vergelijken met managementoverzichten bij de Muziekwinkel.



Figuur 8.13 Menustructuur project Schaatstoernooi2

Uitwerkingen hoofdstuk 9

9.1 De regel 'Elk bedrijf betaalt zijn managers een salaris dat ten minste gelijk is aan een zeker minimumsalaris' volgt niet uit het informatiediagram. De attribuutnaam 'minimumsalaris' suggereert dat misschien, maar dwingt niets af. Verder vermeldt de toelichting dat een commissariaat altijd wordt vervuld voor een *ander* bedrijf. Het diagram sluit echter commissariaten voor het eigen bedrijf niet uit.

9.2 De expressie luidt:

```
b.managers.commissariaten.bedrijf
```

9.3 Formulering in RuleSpeak: 'Een directeur mag alleen aan een andere directeur van het eigen bedrijf rapporteren.'

Opmerking: vanwege bedrijfsregels 3 en 5 kan deze regel worden vereenvoudigd tot: 'Een directeur mag alleen aan een directeur rapporteren.' Hiermee wordt bereikt dat de regels onafhankelijk zijn van elkaar. In paragraaf 9.4.8 wordt ingegaan op onderlinge afhankelijkheid van bedrijfsregels.

9.4 De voorgestelde regel zit al 'ingebakken' in de structuur van het informatiediagram, door de identificatieregel (i) van klasse Commissariaat. Bij onze benadering van bedrijfsregels laten we deze regel buiten beschouwing.

9.5 Regel 6 is een intraklasseregel, de overige regels zijn interklassenregels.

9.6 Omdat we uitgaan van harde handhaving vallen warnings af. Regels 1 t/m 8 schrijven een toestand voor (of verbieden een toestand). Deze kunnen we dus classificeren als rejector. De regels 9 en 10 schrijven het berekenen van een grootheid voor en zijn dus calculators. De regels 11 en 12 zijn weer rejectors.

9.7 Regel 3 is een interklassenregel, omdat via het recursieve associatieattribuut rapporteert_aan naar een tweede exemplaar van de klasse Manager wordt genavigeerd. Omdat twee objecten direct worden vergeleken, is de code echter eenvoudig:

```
context Manager
   inv: self.rapporteert_aan = null or (self <> self.rapporteert_aan)
```

9.8 Een OCL-expressie voor regel 5 luidt:

```
context Manager
   inv: self.bedrijf = self.rapporteert_aan.bedrijf or self.rapporteert_aan = null
```

9.9 Een OCL-expressie voor regel 12 luidt:

```
context Commissariaat
   inv: not(self.bedrijf = self.manager.bedrijf)
```

9.10 Omdat OCL geen operator heeft voor gemiddelde, moeten we het gemiddelde zelf maken met behulp van sum() en size():

```
context Bedrijf::gemiddelde_salaris
   derive: self.managers.salaris -> sum()/size()
```

9.11 Om versie a te vertalen naar OCL moeten we vanuit een contextinstantie van Huis naar alle hypotheken daarvan navigeren en nagaan of de gever van de hypotheek ook de eigenaar is van het huis. We krijgen dan:

```
context Huis
   inv: self.hypotheken.hypotheekgever -> forAll(p | p = self.eigenaar)
```

Om versie b te vertalen naar OCL moeten we de navigatie starten vanuit een contextinstantie (self) van Persoon. Er moet gelden dat de verzameling huizen waarop self een hypotheek heeft, een deelverzameling is van de verzameling huizen waarvan self eigenaar is. Dit leidt tot de volgende formulering met includesAll:

```
context Persoon
   inv: self.huizen -> includesAll(self.hypotheken.huis)
```

9.12 Nee, de bedrijfsregel is geen regel die eisen stelt aan een bevoegdheid. Immers, als je bevoegd bent voor een cursus, impliceert dat verder niets voor het geven van de cursus. Er is dan ook geen herformulering in RuleSpeak of OCL mogelijk met Bevoegdheid als context.

9.13 Bij veranderingen van de bedrijfsregel is aanpassing in het eerste geval eenvoudiger: het is dan niet nodig de structuur van het informatiemodel te wijzigen. Als bijvoorbeeld in geval van schaarste op de arbeidsmarkt de eis van een bevoegdheid wordt losgelaten, dan betekent dit voor het eerste model slechts het verwijderen van een bedrijfsregel met bijbehorend gedrag. Dit is minder ingrijpend dan een structuurverandering.

Daarnaast kunnen we vraagtekens plaatsen vanuit conceptueel standpunt. Het centrale concept van de nieuwe oplossing luidt: 'cursus(aanbieding) die wordt gegeven door een bevoegdheid'. Dit doet geforceerd aan. We houden het liever bij een 'cursus(aanbieding) die wordt gegeven door een persoon, met als eis dat die persoon bevoegd is'.

9.14 De expressies in OCL luiden:

```
context Order
   inv: self.geannuleerd implies not(self.annuleringsdatum = null)
   inv: not(self.geannuleerd) implies self.annuleringsdatum = null)
```

9.15 Een afstanddeelname moet een afstand hebben waarvan het geslacht hetzelfde is als dat van de rijder.

```
context Afstanddeelname
   inv: self.afstand.geslacht = self.toernooideelname.rijder.geslacht
```

9.16 Per rit mag er hoogstens één afstanddeelnemer zijn met startpositie = 'rd' en hoogstens één met startpositie = 'wt'.

```
context Rit
   inv: self.afstanddeelnames -> select(startpositie = 'rd') -> size() <= 1
        and
        self.afstanddeelnames -> select(startpositie = 'wt') -> size() <= 1
```

Eenvoudiger is:

```
context Rit
   inv: self.afstanddeelnames -> isUnique(a | a.startpositie)
```

9.17 We kunnen weer isUnique gebruiken:.

```
context Rit
   inv: self.afstanddeelnames -> isUnique(a | a.toernooideelname)
```

9.18a Herformulering: een afstanddeelnemer mag alleen aan een rit op een bepaalde afstand deelnemen wanneer deze voor de afstand van die rit is ingeschreven.

b OCL-code:

```
context Afstanddeelname
   inv: self.afstand = self.rit.afstand
```

9.19 OCL-code:

```
context Afstandrecord
   inv: self.recordtype = 'W'
        implies self.afstand.afstandsrecords
                -> forAll(a | a.tijd_punten >= self.tijd_punten)
```

Uitwerkingen hoofdstuk 10

10.1 De code voor de after delete trigger luidt aldus:

```
1     create trigger tCalcTotaal_vergoedingen_ad
      active after delete on Vergoeding
      as
2     begin
3       update Manager M
        set    M.totaal_vergoedingen = (select sum(bedrag)
                                        from   Vergoeding V
                                        where  V.manager = M.nr)
        where  M.nr = old.manager);
4     end
```

Toelichting
– In de triggernaam komt tot uiting dat dit een after deletetrigger is: tCalcTotaal_vergoedingen_ad.
– De code is bijna gelijk aan die van tCalcTotaal_vergoedingen_ai. Herberekening vindt nu alleen plaats voor de manager van de zojuist verwijderde Vergoeding-rij (`old.manager`).

10.2 De sum levert in dat geval de waarde null op, omdat de tabel waarover gesommeerd wordt geen records bezit voor de betreffende manager. Kolom totaal_vergoedingen is als not null gedefinieerd en de delete wordt niet geaccepteerd. De meeste SQL-dialecten kennen een functie coalesce(arg1, arg2, arg3…), die de waarde van het eerste argument oplevert dat geen null-waarde heeft. De aangepaste code luidt:

```
update Manager M
set M.totaal_vergoedingen = (select coalesce(sum(V.bedrag),0)
                             from   Vergoeding V
                             where  V.manager = M.nr)
```

10.3 Creëer eerst een exception:

```
create exception eRapportageAanZichzelf
   'Een manager mag niet aan zichzelf rapporteren';
```

Triggermomenten (zie stap 6 stappenplan):

tabel	*event*
Manager	before insert
Manager	before update

De triggercode wordt:

```
create trigger tRapportage_biu
before insert or update on Manager
as
begin
  if (not(new.rapporteert_aan is null or new.rapporteert_aan <> new.nr))
     then
     exception eRapportageAanZichzelf;
end;
```

10.4 Het attribuut aantal_managers van Bedrijf kan alleen veranderen bij een insert of een delete op Manager. (In de applicatie staan we niet toe dat het i-attribuut Manager.bedrijf wordt veranderd.)

Het SQL-statement wordt:

```
update Bedrijf B
set    B.aantal_managers = (select count(nr)
                            from   Manager M
                            where  M.bedrijf = B.naam)
```

De events met beperkingen:

dataset	*event*	*beperking*
Manager	after insert	B.naam = new.bedrijf
Manager	after delete	B.naam = old.bedrijf

10.5 Negative SQL-statement:

```
select 'fout'
from   Manager M
where  M.is_directeur = 'J' and M.salaris < 5000
```

De events met beperkingen:

dataset	*event*	*beperking*
Manager	after insert	M.nr = new.nr
Manager	after update	M.nr = new.nr

10.6 Het negative SQL-statement wordt:

```
select 'fout'
from   Commissariaat C
       join Manager M on C.manager = M.nr
       join Bedrijf B on C.bedrijf = B.naam
where  M.bedrijf = C.bedrijf
```

De events met beperkingen:

dataset	*event*	*beperking*
Manager	after update	M.nr = new.nr
Commissariaat	after update	M.nr = new.manager

Opmerking: updates van Commissariaat zijn verboden, omdat Commissariaat alleen i-attributen bevat (zie paragraaf 2.4)

10.7 De regel moet worden gecontroleerd bij een (poging tot) insert van Afstanddeelname. We maken dus een before insert trigger voor tabel Afstanddeelname.

We zullen gebruikmaken van twee lokale variabelen: geslachtRijder en geslachtAfstand. We vragen die met behulp van een query op en vergelijken ze. Als ze ongelijk zijn, gooien we een exception op.

De code luidt:

```
create exception eVerschillendGeslacht 'Het geslacht van een afstanddeelnemer moet gelijk zijn aan het geslacht van
de verreden afstand.';

create trigger tCheckGeslacht_bi
active before insert on Afstanddeelname
as
declare variable geslachtRijder char(1);
declare variable geslachtAfstand char(1);
begin
  select geslacht
  from   Rijder
  where  nr = new.toernooideelname
  into   :geslachtRijder;

  select geslacht
  from   afstand
  where  id = new.afstand
  into   :geslachtAfstand;

  if (not(geslachtRijder = geslachtAfstand)) then
    exception eVerschillendGeslacht;
end
```

10.8 Voor de rule engine kunnen we de volgende negative select opstellen. We selecteren de 'foute ritten':

```
select 'fout'
from   Rit R
       join Afstanddeelname Ad on R.id = Ad.rit
where  not(R.afstand = Ad.afstand)
```

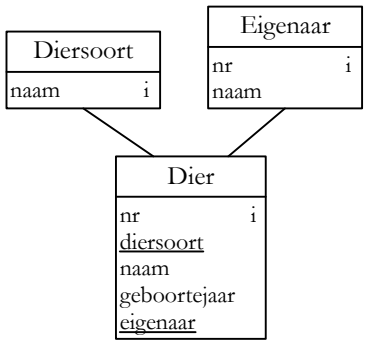
Voor de events en beperkingen hebben we de volgende tabel:

dataset	*event*	*beperking*
Afstanddeelname	after insert	`Ad.id = new.id`

Uitwerkingen hoofdstuk 11

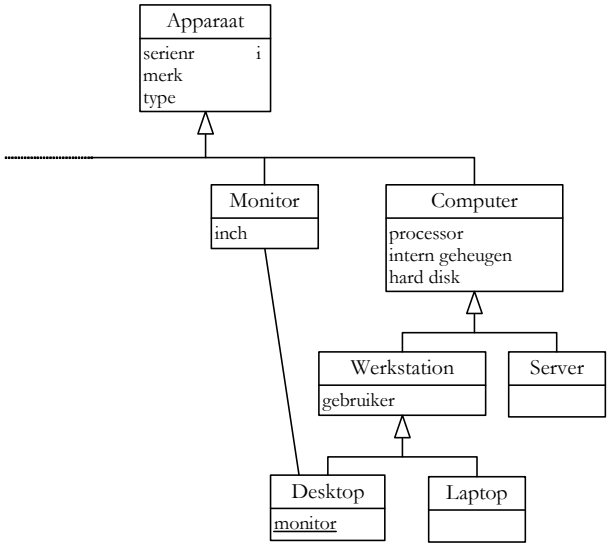
11.1a De subklassen in het model van figuur 11.4 hebben geen eigen attributen en evenmin eigen associaties. Het voordeel van de generalisatiestructuur voor de applicatie is daarom twijfelachtig.

b Het veel simpeler model van figuur 11.39 zal een eenvoudiger informatiesysteem opleveren. En nog rijker ook, in de zin dat de diersoorten niet zijn beperkt tot katten, honden, knaagdieren en 'andere dieren'.



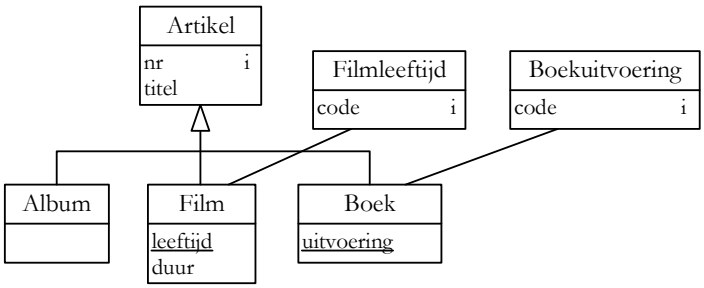
Figuur 11.39 Verbeterd model voor dierenartspraktijk

11.2 Zie figuur 11.40. Monitoren zijn hierin gestandaardiseerd tot een aparte klasse, subklasse van Apparaat. Merk op dat het model onder meer uitdrukt dat een desktop een computer *is*, maar een monitor *heeft*. Het lijntje tussen Desktop en Monitor drukt dus een associatie uit (let ook op de onderstreping van het attribuut monitor), alle overige lijnen drukken een generalisatierelatie uit.



Figuur 11.40 Desktop is werkstation en heeft monitor

11.3 De generalisatiestructuur van figuur 11.17 moet worden uitgebreid met standaardisatieklassen voor Leeftijd en Uitvoering. De corresponderende attributen worden associatieattributen (zie figuur 11.41).



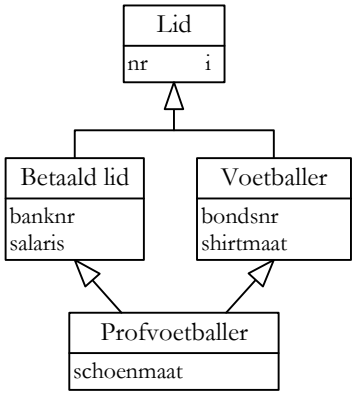
Figuur 11.41 Standaardisatie van filmleeftijden en boekuitvoeringen

Opmerking: buiten de context van Film respectievelijk Boek zijn 'leeftijd' en 'uitvoering' niet erg duidelijk. De standaardisatieklassen hebben we daarom Filmleeftijd en Boekuitvoering genoemd.

11.4a Zonder nadere informatie weten we niet tot welk subklasse (Album, Film of Boek) artikel 1 behoort. De nieuwe structuur staat toe alle rollen (uit Rol) toe te kennen aan elk artikel. Het is wel aannemelijk dat het om een film gaat, maar voor die conclusie moeten we op de eindgebruiker vertrouwen. De applicatie geeft (indien geen aanvullende constraints zijn gedefinieerd) geen enkele garantie.

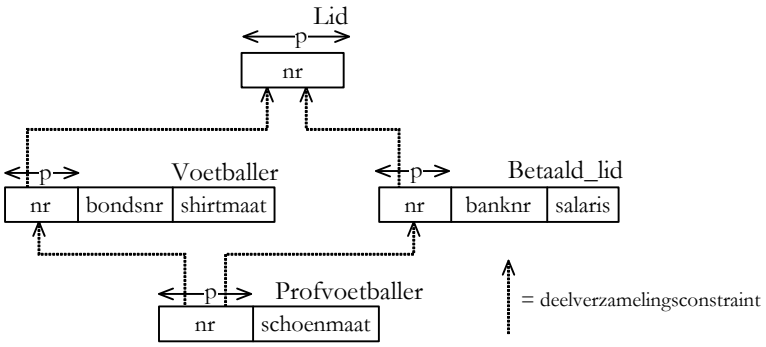
b Ja, zonder aanvullende constraints kan dat.

11.5 Omdat aan 'voetballend' extra attributen zijn verbonden, maken we een subklasse Voetballer (zie figuur 11.42). Omdat dat ook geldt voor 'betaald', maken we een tweede subklasse 'Betaald lid'. Een instantie van Lid kan wel of niet tot de subklasse 'Betaald lid' behoren. Onafhankelijk daarvan kan de instantie wel of niet tot de subklasse Voetballer behoren. De generalisatie is dus niet-exclusief en niet-totaal. Profvoetballers horen tot beide subklassen. Dit geeft een klasse Profvoetballer, die een gemeenschappelijke subklasse is van 'Betaald lid' en Voetballer. De klasse Profvoetballer erft alle attributen van de klassen hoger in de generalisatiehiërarchie.



Figuur 11.42 Generalisatiestructuur voetbalclub

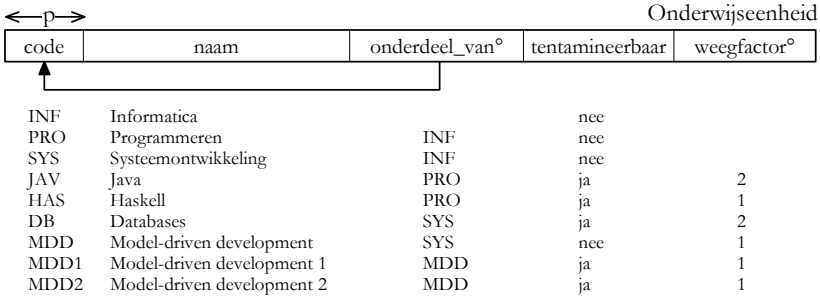
11.6a Zie figuur 11.43.



Figuur 11.43 Subklassetabel Profvoetballer met twee deelverzamelingsconstraints

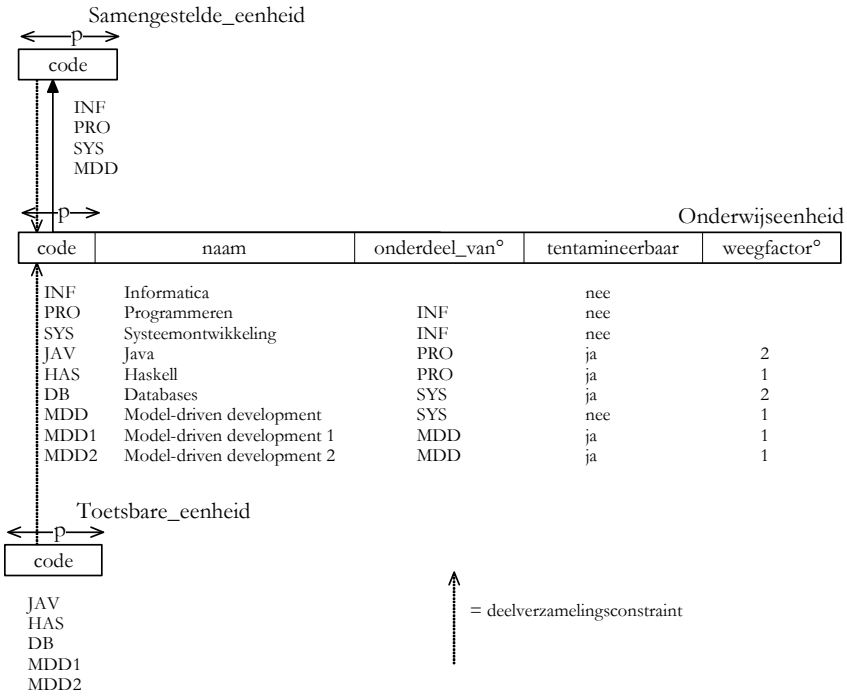
b Bij een verwijssleutel hoort altijd één primaire sleutel (van 'de' oudertabel). Als de deelverzamelingsconstraints verwijssleutels waren, dan zou Profvoetballer.nr in figuur 11.43 een verwijssleutelkolom zijn met twee oudertabellen, wat onmogelijk is.

11.7a Het resultaat komt overeen met de uitwerking van opgave 6.10b, zie figuur 11.44.



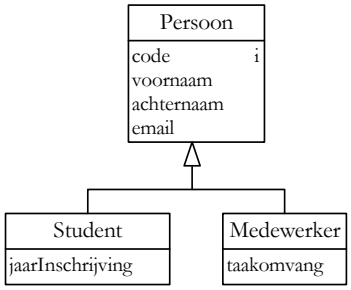
Figuur 11.44 PSM bij figuur 11.37 (transformatie naar één tabel)

b Het PSM omvat nu drie tabellen met deelverzamelingsrelaties voor de primaire sleutels (zie figuur 11.45). Omdat de subklasse SamengesteldeEenheid de ouderrol vervult bij de associatie, hebben we die bovenaan getekend.



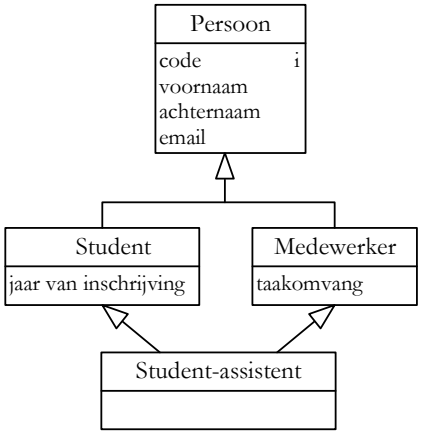
Figuur 11.45 PSM bij figuur 11.37 (transformatie naar aparte superklasse- en subklassetabellen

11.8 Aangezien niemand zowel student als medewerker kan zijn, kunnen studentnummer en acroniem worden gegeneraliseerd tot een identificerende persoonscode. Deze en de andere gemeenschappelijke attributen kunnen nu worden ondergebracht in een superklasse Persoon. De specifieke attributen brengen we onder in subklassen Student en Medewerker, die elkaar uitsluiten (exclusieve generalisatie). Zie figuur 11.46).



Figuur 11.46 Studenten en medewerkers: exclusieve generalisatie

11.9 Het probleem is dat student-assistenten als student een studentnummer hebben en er als medewerker een acroniem bij krijgen. Dit roept de vraag op welke van beide gebruikt zal worden ter identificatie. Of zal dat van de situatie afhangen? Neem als voorbeeld de gebruikerstoegang voor het netwerk: krijgt de student-assistent twee accounts, een studentaccount en een medewerkeraccount? Dit lijkt nutteloos en zelfs onwenselijk. We nemen aan dat er een keuze gemaakt moet worden. Voor de hand ligt de student-assistent, die al een studentnummer had, geen acroniem te geven. De oplossing van figuur 11.47 kan dan in essentie worden gehandhaafd en worden uitgebreid met een subklasse Student-assistent, die zowel Student als Medewerker als superklasse heeft (zie figuur 11.47).



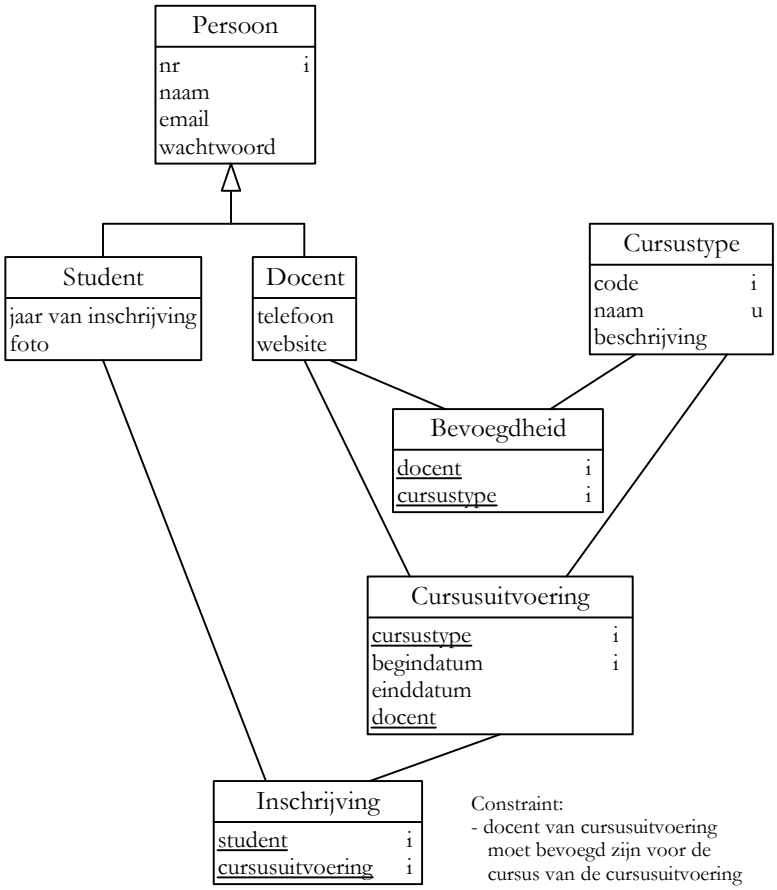
Figuur 11.47 Een student-assistent is zowel student als medewerker

11.10a De term 'cursus' wordt in twee betekenissen gebruikt (homoniemen): in de betekenis van cursustype (onafhankelijk van een specifieke uitvoering) en in de betekenis van cursusuitvoering (vanaf een specifieke datum).
De term cursus wordt enkele keren gebruikt synoniem aan cursusuitvoering.

b Het valt nog helemaal niet mee de tekst te herschrijven zonder homoniemen of synoniemen. We hebben ervoor gekozen het woord 'cursus' maar helemaal te vermijden.
Een cursusinstituut verzorgt cursusuitvoeringen. Elke cursusuitvoering hoort tot een cursustype. Een cursusuitvoering wordt gekenmerkt door het cursustype en een specifieke begindatum. Een cursusuitvoering wordt verzorgd door een docent. Een docent mag een cursusuitvoering alleen verzorgen wanneer die voor het cursustype van die cursusuitvoering bevoegd is. Studenten die een cursusuitvoering willen volgen, moeten zich daarvoor inschrijven.

c Na de exercitie van onderdeel b is het voornaamste werk gedaan. Onontkoombaar is het exemplaarpatroon Cursustype-Cursusuitvoering. Verder ligt generalisatie voor de hand van Student en Docent tot Persoon. Het resultaat, mede gebaseerd op aanvullende aannames over kenmerken van studenten, docenten, cursustypen en cursusuitvoeringen, ziet u in figuur 11.48.

Naamgeving: in plaats van Cursustype-Cursusuitvoering kunt u ook kiezen voor Cursus-Cursusuitvoering of Cursustype-Cursus. Het exemplaarpatroon gaat vaak samen met homoniem spraakgebruik!



Figuur 11.48 Informatiemodel cursusinstituut

De eis dat een docent die een cursus (!) verzorgt, bevoegd moet zijn voor die cursus (!), kunnen we niet vangen in de structuur. Die eis is als constraint toegevoegd en zal via code moeten worden geïmplementeerd.

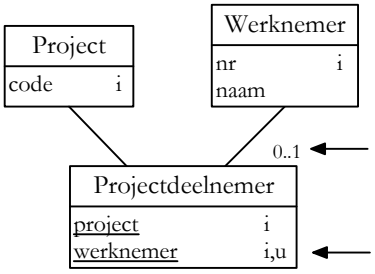
Het model van figuur 11.48 is als voorbeeldmodel beschikbaar, onder de naam Cursusinstituut.

Uitwerkingen hoofdstuk 12

12.1a De informatiediagrammen zijn equivalent wanneer we de aanvullende multipliciteitsconstraint in aanmerking nemen. Immers, beide diagrammen staan toe projecten (met een code) en werknemers (met een nummer en een naam) op te nemen en beide staan toe maximaal één project per werknemer te registreren.

b Informatiestructuur b is het meest generiek. Bij een verandering in de projectorganisatie waarbij meerdere projecten per werknemer worden toegestaan, hoeft in diagram b alleen de constraint te worden geschrapt. Diagram a vereist een structuurverandering.

c De 0..1-constraint is equivalent met een uniciteitsregel voor Projectdeelnemer.werknemer (zie figuur 12.10).



Figuur 12.10 Multipliciteitsconstraint: equivalent met uniciteitsconstraint

d Zie het antwoord op onderdeel c. Het schrappen van een constraint is een veel eenvoudiger ingreep dan een structuurverandering.

e Deze vraag is moeilijk te beantwoorden. Wanneer men zeker weet dat de één-projectregel stabiel is, is er niets op tegen deze in de structuur te verankeren. Maar als versoepeling van de regel goed denkbaar is, doet men er goed aan voor structuur b te kiezen.

f Deze oplossing is zeker te overwegen. Het is helemaal niet altijd nodig een bedrijfsregel hard af te dwingen. Er zijn in een bedrijf nog andere regelmechanismen dan harde systeemconstraints. Van harde constraints heeft men soms alleen maar last. Het kan gewenst zijn overtredingen wel te signaleren maar tijdelijk toe te staan.

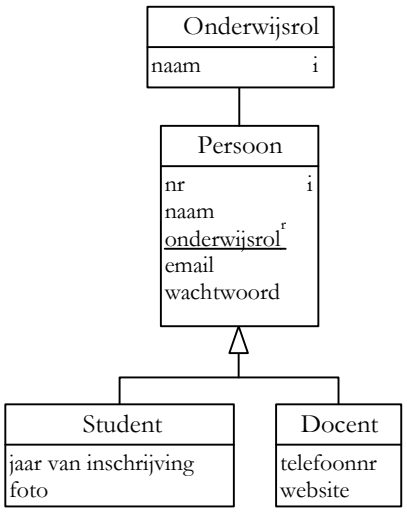
12.2 Deze kwestie is niet met een eenvoudig 'ja' of 'nee' te beantwoorden. We komen er op terug in paragraaf 12.2.6.

12.3a Indien we in een superklasseformulier alle attributen van de subklassen opnemen, moeten we weten of gelijknamige subklasseattributen semantisch gelijk zijn. Zo ja, dan moet er in het formulier één veld komen voor beide attributen, zo nee, dan moet elk attribuut een eigen veld krijgen (met een onderscheidend label).

b Het probleem doet zich volledig analoog voor wanneer bij de PIM-PSM-transformatie gelijknamige subklasseattributen worden opgenomen in één tabel voor de hele generalisatiehiërarchie. Wanneer de gelijknamige attributen semantisch gelijk zijn, volstaat één kolom. In het andere geval zijn twee kolommen nodig en moet de naamgeving worden aangepast.

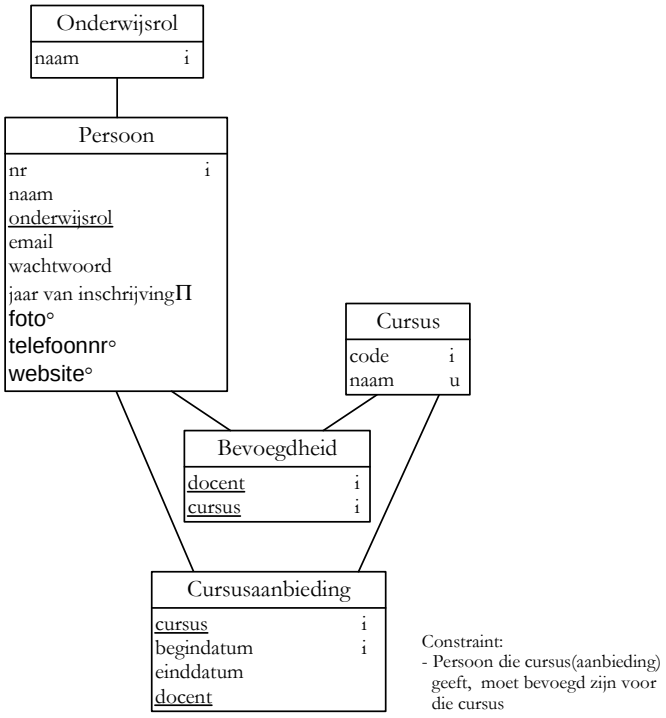
12.4 Ja, dat maakt verschil.
Indien de tool gelijknamige attributen als semantisch gelijk behandelt, kan de ontwikkelaar verschillende namen kiezen voor semantisch ongelijke attributen.
Als de tool gelijknamige attributen als semantisch ongelijk behandelt, heeft de ontwikkelaar een probleem wanneer deze van mening is dat de attributen semantisch gelijk zijn en dus moeten samensmelten in bijvoorbeeld een superklasseformulier.

12.5a Als eerste stap modelleren we de generalisatie van figuur 12.9 uit, vanuit het gegeven dat de generalisatie exclusief is en totaal. De superklasse krijgt dan een verplicht subklassebepalend attribuut dat de soort persoon aangeeft (student of docent). In plaats van 'soort' kiezen we voor 'onderwijsrol': een persoon heeft de rol student of de rol docent. Zie figuur 12.11.



Figuur 12.11 Generalisatie uitgemodelleerd via onderwijsrollen

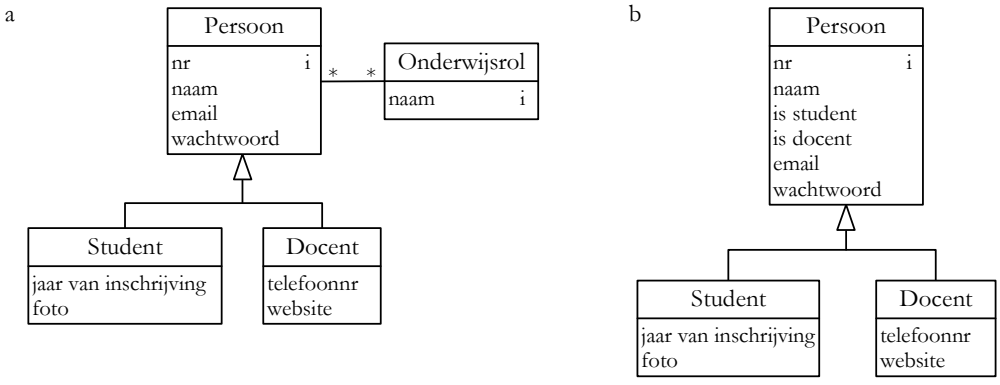
Degeneralisatie houdt nu in het schrappen van de subklassen. Zie figuur 12.12 voor het nieuwe diagram.



Figuur 12.12 Structuur na degeneralisatie

b De kosten van de transformatie zijn als volgt:
- Student- en docentformulieren zijn nu non-default.
- Bevoegdheid en Cursusaanbieding zijn nu geassocieerd met Persoon, zodat een constraint nodig is om deze te beperken tot docenten (personen met de onderwijsrol 'docent').

12.6 Figuur 12.13 geeft twee alternatieven voor de uitgemodelleerde generalisatie. Figuur a maakt gebruik van een n-op-m associatie tussen personen en onderwijsrollen. Figuur b modelleert de niet-exclusiviteit via de boolean attributen 'is student' en 'is docent'. Er geldt de constraint dat ten minste één van de boolean attributen true is.



Figuur 12.13 Niet-exclusieve generalisatie (uitgemodelleerd)