Tentamen Gegevensbanken (211074) — 29 oktober 2009

CONTROLEER EERST OF ALLE BLADZIJDEN T/M BLZ. 15 AANWEZIG ZIJN!

Vul het tentamenbriefje volledig in, zódanig dat BEIDE DOORSLAGEN goed leesbaar zijn.

NAAM, VOORLETTERS: _____

STUDENTNUMMER: _____

OPLEIDING: _____

De uitwerkingen moeten op deze opgavenformulieren worden genoteerd in de daarvoor bestemde vakken. Alle overige ruimte kun je zo nodig als **kladpapier** gebruiken en wordt niet bekeken en niet beoordeeld.

Het gebruik van boeken, dictaten en dergelijke is *niet* toegestaan, behoudens één vel van A4 formaat met *eigen* aantekeningen (dubbelzijdig, getypt of geschreven) of kopieën van *delen van het boek*; kopieen van ander materiaal (zoals tentamenuitwerkingen) zijn niet toegestaan.

Normering: per opgave staan de te behalen punten in de kantlijn en u krijgt 5 punten gratis; samen zijn dat 100 punten. Het eindcijfer is het aantal behaalde punten gedeeld door 10. Onleesbare tekst wordt steeds fout gerekend.

Na afloop moet de *volledige* set opgavenformulieren worden ingeleverd; het kladpapier niet. De tentamenopgaven zijn niet geheim en worden voorzien van modeluitwerkingen op Blackboard gepubliceerd. (Die modeluitwerkingen moet je op papier of elektronisch bij je hebben wanneer je je tentamen komt inzien.)

Niet invullen

5 gratis	1	2	3	3bonus	4	5	6	7	8	9	10	10bonus

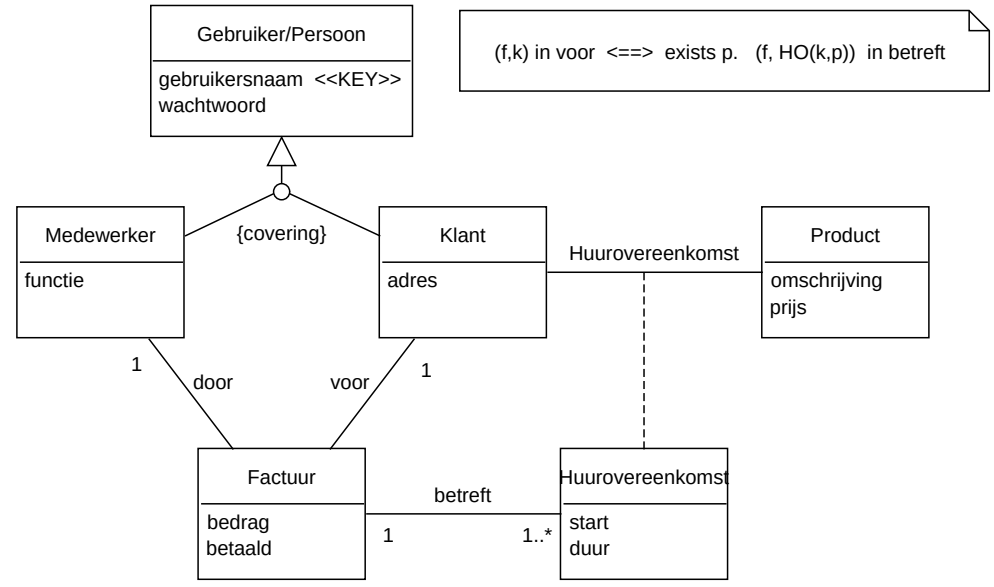
10p. **Opgave 1.** Een verhuurbedrijf kent medewerkers, klanten en producten.

- Zowel medewerkers als ook klanten hebben toegang tot het informatiesysteem van het bedrijf middels een (unieke) gebruikersnaam en daaraan gekoppeld wachtwoord.
- Klanten kunnen producten huren, in andere woorden, producten kunnen aan klanten verhuurd worden; zo'n huur wordt in een huurovereenkomst vastgelegd en heeft een startdatum en duur.
- Voor iedere huurovereenkomst wordt een factuur uitgeschreven; een factuur is uitgeschreven door één medewerker en betreft één klant en één of meer huurovereenkomsten. Per factuur is bekend wat het bedrag is en of de factuur betaald is.
- Verschillende personen (dat wil zeggen, medewerkers en klanten) hebben verschillende gebruikersnamen.
- Van iedere klant is het adres bekend. Van iedere medewerker is de functie bekend.
- Een product heeft een omschrijving en een prijs.

Geef in het antwoordblok een Entity-Relationship diagram dat de gegevens voor één tijdstip *zo precies mogelijk* modelleert en gebruik daarbij *zo geschikt mogelijke* constructies. *Doe het eerst op kladpapier en dan pas in het net.* Zowel de ERD-notatie uit het boek als ook de UML notatie (class diagram) is toegestaan, *maar een mengeling van beide niet.*

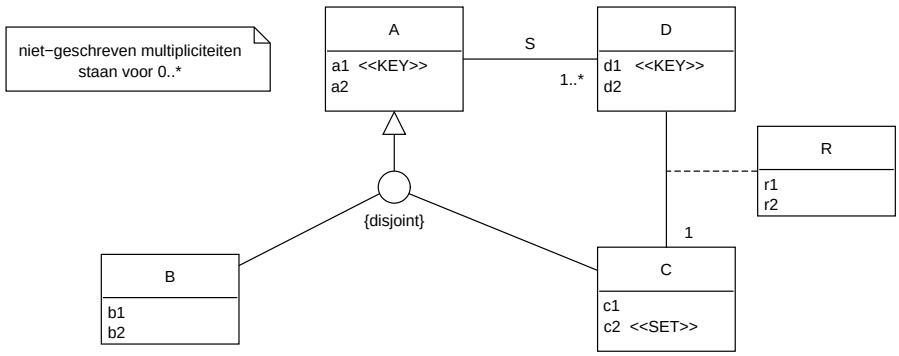
Onleesbare tekst wordt fout gerekend.

Er zijn verschillende mogelijkheden; zie de toelichting.



(Zie Toelichting.)

10p. **Opgave 2.** Beschouw het volgende ER-diagram in de notatie van de UML:



Het ERD kan op verschillende manieren vertaald worden naar een databaseschema dat geschikt is om informatie die past in het ERD op te slaan. U dient een manier te kiezen waarbij: (1) er *zo weinig mogelijk* relatieschema's in het databaseschema zijn, maar met de beperkingen dat (2) er geen NULLs nodig zijn vanwege de vertaling, (3) er geen redundantie geïntroduceerd wordt door de vertaling, en (4) alle attributen atomaire waarden hebben.

Geef de relatieschema's in SQL syntaxis waarbij de tekst '*create table*' en iedere domeinindicatie weggelaten mag worden; een voorbeeld van de vorm van zo'n schema is:

$X(x_1, \ldots,$ primary key $(x_i, x_j ...),$ foreign key $(x_m, x_n ...)$ references $Y(y_1, y_2, ...), \ \ldots)$

(U mag "primary key" afkorten tot "PK", en zo voorts, en er mogen CHECKs voorkomen.)

Onleesbare tekst wordt fout gerekend.

$A(a1, a2,$
primary key $(a1)$, check (a1 in select $a1$ from S))
$B(a1, b1, b2,$
primary key $(a1)$, foreign key $(a1)$ references $A(a1)$,
check ($a1$ not in (select $a1$ from C)))
$C(a1, c1,$
primary key $(a1)$, foreign key $(a1)$ references $A(a1)$,
check ($a1$ not in (select $a1$ from B)))
$C1(a1, c2',$
primary key $(a1, c2')$, foreign key $(a1)$ references $C(a1)$)
$D(d1, d2, a1, r1, r2,$
primary key $(d1)$, foreign key $(a1)$ references $C(a1)$, $a1$ not null)
$S(a1, d1,$
primary key $(a1, d1)$,
foreign key $(a1)$ references $A(a1)$, foreign key $(d1)$ references $D(d1)$)
(Zie Toelichting.)

10p. **Opgave 3.** Beschouw de wereld van huizen en makelaars zoals gezien door een makelaarskantoor: er zijn entiteiten en attributen zoals huis, makelaar, adres, bouwjaar, vraagprijs, kenmerk (namelijk: oppervlakte, inhoud, ligging), eigenaar, telefoonnummer, klant (van het kantoor), specialiteit (van een makelaar). Deze vormen samen als volgt een relatie $\mathcal{R}$.

Een tuple $(H, B, V, K, E, T, N, M, S)$ zit op tijdstip t in $\mathcal{R}$ precies wanneer al het volgende geldt op tijdstip t:

1. H is een Huis
2. B is het Bouwjaar van huis H
3. V is de Vraagprijs van huis H
4. K is één van de Kenmerken (oppervlakte, inhoud, ligging, ...) van huis H
5. E is de Eigenaar van huis H
6. T is één van de Telefoonnummers (mobiel, thuis, ...) van eigenaar E
7. N is de aanduiding of eigenaar E een Nieuwe klant is bij het kantoor
8. M is de Makelaar die de verkoop van huis H begeleidt
9. S is één van de Specialiteiten van makelaar M (bungalows, villa's, ...)

Geef voor ieder van de functionele afhankelijkheden hieronder, met een letter W aan of die volgens bovenstaande definitie altijd Waar is in relatie $\mathcal{R}$, en met een letter O of die afhankelijkheid volgens bovenstaande definitie Onwaar kan zijn in relatie $\mathcal{R}$, en motiveer kort uw keuze (de motivatie telt mee in de beoordeling).

Onleesbare tekst wordt fout gerekend.

FD	W/O	motivatie bij keuze
$H \to B$	W	een huis heeft hooguit één bouwjaar (regel 2)
$B \to V$	O	bij één bouwjaar kunnen verscheidene (huizen met) vraagprijzen horen
$BV \to H$	O	verscheidene huizen kunnen eenzelfde bouwjaar en vraagprijs hebben
$H \to K$	O	een huis kan verscheidene kenmerken hebben (regel 4)
$E \to N$	W	een eigenaar heeft hooguit één aanduiding 'wel/niet nieuw' (regel 7)
$N \to H$	O	verschillende klanten (dus huizen) kunnen dezelfde N-aanduiding hebben
$H \to N$	W	vanwege $H \to E$ (5), en $E \to N$ (7)
$HS \to M$	W	vanwege $H \to M$ (8) en augmentatie
$HM \to S$	O	een makelaar kan meerdere Spec. hebben (9), zelfs als H bekend is *(Zie Toelichting.)*

Geef voor ieder van de volgende MultiValued Dependencies (MVD) aan of die *W*aar of *O*nwaar is in de tabel $\mathcal{R}$:

Onleesbare tekst wordt fout gerekend.

MVD	W/O
1. $HBVKETNMS = HBVKETNM \bowtie MS$	W
2. $HBVKETNMS = HBVK \bowtie ETN \bowtie MS$	O
3. $HBVKETNMS = HB \bowtie HV \bowtie HK \bowtie HE \bowtie ET \bowtie EN \bowtie HM \bowtie MS$	W

Het volgende onderdeel hoeft niet beantwoord te worden, maar bij goede beantwoording krijgt u maximaal 5 bonuspunten (boven op de punten die u voor deze opgave kunt verdienen). Geef een motivatie voor de keuzen W/O hierboven:

Onleesbare tekst wordt fout gerekend.

1. Uit de definitie van $\mathcal{R}$ volgt voor alle $(h, b, v, k, e, t, n, m) \in \pi_{HBVKETNM}\mathcal{R}$:

$\forall\, s \mid (m, s) \in \pi_{MS}\mathcal{R} \bullet (h, b, v, k, e, t, n, m, s) \in \mathcal{R}$.

Dus $\pi_{HBVKETNM}\mathcal{R} \bowtie \pi_{MS}\mathcal{R}$ levert niet méér rijen dan er in $\mathcal{R}$ zitten: de join is lossless.

2. Onwaar omdat o.a. het verband H–E verloren gaat.

De join van de rhs levert HE-combinaties die in de lhs niet bestaan.

3. $HB \bowtie HV \bowtie HK \bowtie HE \bowtie ET \bowtie EN \bowtie HM \bowtie MS$

$= (((((HB \bowtie HV \bowtie HE \bowtie HM) \bowtie EN) \bowtie HK) \bowtie ET) \bowtie MS)$ ass. en comm. van $\bowtie$

$= ((((HBVEM \bowtie EN) \bowtie HK) \bowtie ET) \bowtie MS)$ wegens (a)

$= (((HBVEMN \bowtie HK) \bowtie ET) \bowtie MS)$ wegens (b)

$= HBVEMKN \bowtie ET) \bowtie MS)$ wegens (c)

$= HBVEMKNT \bowtie MS$ wegens (d)

$= HBVKEMNTS$, wegens (e)

waarbij

(a): De intersectie van de componenten in $HB \bowtie HV \bowtie HE \bowtie HM$ is H, en dat is een key in alle of op-één-na alle componenten ervan.

(b): De intersectie van $HBVEM$ en EN is E, en dat is een key in EN.

(c): Net als bij (1): $HBVENM \bowtie HK = HBVENMK$.

(d): Net als bij (1): $HBVENMK \bowtie ET = HBVENMKT$.

(e): Net als bij (1): $HBVENMKT \bowtie MS = HBVENMKTS$. *(Zie Toelichting.)*

10p. **Opgave 4.** Voor de administratie van een huisartsenpraktijk ziet een deel van het databaseschema er als volgt uit:

Patient (*patientnr*, *adres*, *naam*, ...)
Factuur (*patientnr*, *datum*, *bedrag*, ...)

Een *patientnr* identificeert een patiënt uniek, en er worden in de administratie geen *patientnr*s gebruikt die niet in *Patient* voorkomen. Een factuur wordt uniek geïdentificeerd door een *patientnr* en *datum* samen. We willen dat iedere wijziging van de databasetoestand zich houdt aan de volgende regels:

(a) De gegevens van een patiënt worden niet uit *Patient* verwijderd als zijn *patientnr* nog in *Factuur* voorkomt.

(b) Het *patientnr* in *Factuur* verandert op gelijke wijze als in *Patient* wanneer in *Patient* een wijziging wordt aangebracht.

Voeg key constraints toe aan het schema zó dat wijzigingen van de databasetoestand zich automatisch houden aan bovenstaande regels; voeg geen overbodige constraints toe.
Ter herinnering, key constraints zien er als volgt uit:

primary key $(x, x', \ldots)$,
foreign key $(x, x', ...)$ references $T(y, y', ...)$ on *event action* on *event′ action′* ...

Hierbij staat x voor een attribuutnaam, T voor een tabelnaam, *event* voor een gebeurtenis (*delete*, *update*) en *action* voor een actie (*set null*, *set default*, *no action*, *cascade*).

Onleesbare tekst wordt fout gerekend.

Patient (*patientnr*, *adres*, *naam*, ...,
    primary key (*patientnr*))

Factuur (*patientnr*, *datum*, *bedrag*, ...
-- primary key (...), -- onnodige constraint en niet gevraagd
    foreign key (*patientnr*) references *Patient* (*patientnr*)
        on delete no action -- mag desgewenst weggelaten worden, dit is de default
        on update cascade) *(Zie Toelichting.)*

In plaats van key constraints toe te voegen kan de wens ook automatisch door het database-systeem gerealiseerd worden door geschikte triggers toe te voegen aan het databaseschema. Ter herinnering, de syntaxis van een trigger creatie luidt als volgt:

create trigger *trigger-name*
{before | after} {insert | delete | update [of *column-name-list*] } on *table-name*
[referencing [old as *var*][new as *var*] [old table as *var*][new table as *var*]]
[for each { row | statement }]
[when (*precondition*)]
statement-list

Hierbij staat $\{x \mid y \mid \ldots\}$ voor "een keuze uit $x, y, \ldots$"; en $[x]$ staat voor "optioneel x".

Geef de trigger create statement voor regel (a) van de opgave:

Onleesbare tekst wordt fout gerekend.

create trigger *regel_a*
before delete on *Patient*
referencing old as *O*
for each row
when (*O.patientnr* in (select *patientnr* from *Factuur*))
rollback

Geef de trigger create statement voor regel (b) van de opgave:

Onleesbare tekst wordt fout gerekend.

create trigger *regel_b*
after update of *patientnr* on *Patient*
referencing old as *O* new as *N*
for each row
update *Factuur F*
set *F.patientnr* = *N.patientnr*
where *F.patientnr* = *O.patientnr*

10p. **Opgave 5.** Beschouw het relatieschema $\mathbf{R} = (\bar{R}, \mathcal{F})$, waarbij de attribuutverzameling $\bar{R}$ en de verzameling $\mathcal{F}$ van functionele afhankelijkheden als volgt luiden:

$$\begin{aligned} \bar{R} &= ABCDE \\ \mathcal{F} &= \{AD \to B, \quad B \to C, \quad C \to A, \quad D \to E\} \end{aligned}$$

(1) Geef in het antwoordblok in iedere genummerde regel een zo groot mogelijk rechterlid Y zó dat de functionele afhankelijkheid $X \to Y$ volgt uit de hierboven gegeven verzameling $\mathcal{F}$ (met andere woorden: Y is de closure $X^+_{\mathcal{F}}$). Neem de attributen die al in het linkerlid X staan *niet* in het rechterlid Y op (en schrijf niets op als Y de lege verzameling attributen is).
(2) *Omcirkel* in het antwoordblok de aanwezige *sleutels* van **R**.
(3) *Onderstreep* in het antwoordblok de aanwezige *supersleutels* van **R**.
(4) Omcirkel in het antwoordblok de *nummers* van de aanwezige functionele afhankelijkheden die een schending vormen van de BCNF-eis.

Onleesbare tekst wordt fout gerekend.

	X	$\to$	Y
1		$\to$	
2	A	$\to$	
3	B	$\to$	AC
4	C	$\to$	A
5	D	$\to$	E
6	E	$\to$	
7	AB	$\to$	C
8	AC	$\to$	
9	AD	$\to$	BCE
10	AE	$\to$	
11	BC	$\to$	A
12	BD	$\to$	ACE
13	BE	$\to$	AC
14	CD	$\to$	ABE
15	CE	$\to$	A
16	DE	$\to$	
17	ABC	$\to$	
18	ABD	$\to$	CE
19	ABE	$\to$	C
20	ACD	$\to$	BE

(Zie Toelichting.)

10p. **Opgave 6.** Beschrijf wat een dirty read is:

Onleesbare tekst wordt fout gerekend.

Een dirty read is een lees-actie waarbij een waarde gelezen wordt die eigenlijk helemaal niet bestaat in de database-toestanden die ontstaan door seriele uitvoering van de transacties, met name doordat de gelezen waarde later teruggezet wordt door een abort van de transactie die hem schreef.

Zet in onderstaande tabel een merkteken '$\surd$' bij precies iedere combinatie van isolatieniveaus voor transacties T_1 en T_2 waarbij transactie T_1 géén dirty read zal doen wanneer beide transacties gelijktijdig uitgevoerd worden:

Onleesbare tekst wordt fout gerekend.

Op de met '$\surd$' gemerkte niveaus zal T_1 geen dirty reads doen (elders mogelijk wel):

Isolatieniveau voor T_2: $\downarrow$	Isolatieniveau voor T_1: read uncommitted	read committed	repeatable read	serializable
read uncommitted		$\surd$	$\surd$	$\surd$
read committed		$\surd$	$\surd$	$\surd$
repeatable read		$\surd$	$\surd$	$\surd$
serializable		$\surd$	$\surd$	$\surd$

Het niveau voor T_2 doet niet terzake!

Laat x, y, z staan voor drie verschillende tabellen in de database. Definieer voor $i = 1, 2$ een transactie T_i in termen van read operaties $r_i(x), r_i(y), r_i(z)$ en write operaties $w_i(x)$, $w_i(y)$, $w_i(z)$ en beeindigingen $commit_i$ en $abort_i$ (je hoeft niet *al deze* operaties te gebruiken!), en geef een schedule voor een gelijktijdige executie van T_1 en T_2 zódanig dat T_1 een dirty read doet:

Onleesbare tekst wordt fout gerekend.

$T_1 = r_1(x); \ldots$

$T_2 = w_2(x); abort_2$

schedule: $w_2(x); r_1(x); abort_2; \ldots$

10p. **Opgave 7.** Beschouw het relatieschema $\mathbf{R} = (ABCDEF, \mathcal{F})$, waarbij:

$$\mathcal{F} \;=\; \{ABC{\rightarrow}E, \quad CD{\rightarrow}B, \quad E{\rightarrow}D\}$$

Geef de functionele afhankelijkheden in $\mathcal{F}$ die een schending vormen van de BCNF-conditie voor $\mathbf{R}$? Beargumenteer uw antwoord.

Onleesbare tekst wordt fout gerekend.

Alle drie FDs uit $\mathcal{F}$ vormen een schending; bijvoorbeeld voor $ABC{\rightarrow}E$: deze is niet triviaal (dwz, het rechterlid E is niet deel van het linkerlid ABC) en het linkerlid ABC omvat geen sleutel. De enige sleutel van $\mathbf{R}$ is ACF.

Geef een lossless decompositie van $\mathbf{R}$ in precies twéé schema's, zeg $\mathbf{R}_1$ en $\mathbf{R}_2$, zodanig dat $\mathbf{R}_1$ en $\mathbf{R}_2$ samen minstens één schending minder hebben dan $\mathbf{R}$. (Als u het BCNF-algoritme toepast, dan krijgt u na één stap zo'n decompositie.) Verklaar uw werkwijze en geef heel precies aan wat de attributen en functionele afhankelijkheden van $\mathbf{R}_1$ en $\mathbf{R}_2$ zijn.

Onleesbare tekst wordt fout gerekend.

Neem een FD uit $\mathcal{F}$ die de BCNF-conditie schendt; bijvoorbeeld $ABC{\rightarrow}E$. Splits $\bar{R}$ in $\bar{R}_1 = ABCE$ (alle attributen van de FD) en $\bar{R}_2 = ABCDF$ (alles zonder de attributen uit het rechterlid van de FD). Neem $\mathbf{R}_1 = (\bar{R}_1, \mathcal{F}_1)$ en $\mathbf{R}_2 = (\bar{R}_2, \mathcal{F}_2)$, waarbij $\mathcal{F}_i$ een basis is voor de verzameling van FDs uit $\mathcal{F}^+$ waarin uitsluitend attributen van $\bar{R}_i$ voorkomen:

$\mathbf{R}_1 = (ABCE, \{ABC{\rightarrow}E,\ CE{\rightarrow}B\})$

$\mathbf{R}_2 = (ABCDF, \{ABC{\rightarrow}D,\ CD{\rightarrow}B\})$.

($CE{\rightarrow}B$ zit niet in $\mathcal{F}$, maar wel in $\mathcal{F}^+$ en bevat alleen maar attributen uit $\bar{R}_1$.)

(Zie Toelichting.)

Zijn er in bovenstaande decompositiestap van $\mathbf{R}$ naar $\mathbf{R}_1, \mathbf{R}_2$ functionele afhankelijkheden verloren gegaan? Zo ja, geef dan zo'n functionele afhankelijkheid.

Onleesbare tekst wordt fout gerekend.

Afhankelijkheid $E{\rightarrow}D$ zit wel in $\mathcal{F}$ maar niet in $(\mathcal{F}_1 \cup \mathcal{F}_2)^+$, en is dus verloren gegaan.

Voer nu de volgende opdrachten uit:

- Construeer een lossless decompositie van $\mathbf{R}$ tot schema's die ieder in BCNF staan. (U mag gebruik maken van en verwijzen naar de vorige antwoorden.)
- Verklaar iedere stap zodat het voor de corrector duidelijk is hoe u te werk gaat.
- Geef aan of de functionele afhankelijkheden behouden blijven onder de decompositie.

$$\boxed{\mathcal{F} \quad = \quad \{ABC{\rightarrow}E, \quad CD{\rightarrow}B, \quad E{\rightarrow}D\}}$$

Onleesbare tekst wordt fout gerekend.

Let op: $CE{\rightarrow}B$ volgt uit $\mathcal{F}$ en zal dus (bij een correcte redenering) een FD worden van een component met attributen $...BC...E...$, ook al zit D daar niet bij. Net zo voor $ACD{\rightarrow}E$ en $ABC{\rightarrow}D$.

We passen het BCNF-algoritme toe.

- De eerste stap is in de vorige vragen gedaan en levert bovengenoemde decompositie $\{\mathbf{R}_1, \mathbf{R}_2\}$ van $\mathbf{R}$ op.

- We bekijken nu $\mathbf{R}_1 = (ABCE, \{ABC{\rightarrow}E,\ CE{\rightarrow}D\})$.

In $\mathbf{R}_1$ zijn ABC en ACE de sleutels, dus is $ABC{\rightarrow}E$ geen schending van de BCNF-conditie, maar $CE{\rightarrow}D$ wel. We kiezen $CE{\rightarrow}B$ ter eliminatie. Dus splitsen we $\bar{R}_1$ in $\bar{R}_{1a} = BCE$ en $\bar{R}_{2b} = A\not{B}CE = ACE$. Dit levert schema's $\mathbf{R}_{1j} = (\bar{R}_{1j}, \mathcal{F}_{1j})$, waarbij $\mathcal{F}_{1j}$ (een basis voor) de inperking is van $\mathcal{F}^+$ (of $\mathcal{F}_1^+$) tot $\bar{R}_{1j}$. Dus

$\mathbf{R}_{1a} = (BCE, \{CE{\rightarrow}B\})$ en
$\mathbf{R}_{1b} = (ACE, \{\ \})$.

Beide schema's staan in BCNF. De FD $ABC{\rightarrow}E$ van $\mathbf{R}_1$ is verloren gegaan.

- We bekijken nu $\mathbf{R}_2 = (ABCDF, \{ABC{\rightarrow}D,\ CD{\rightarrow}B\})$.

In $\mathbf{R}_2$ zijn $ABC{\rightarrow}D$ en $CD{\rightarrow}B$ beide een schending van de BCNF-conditie; voor $ABC{\rightarrow}D$ is de reden dat die niet-triviaal is en ABC geen sleutel is in $\mathbf{R}_2$ (want $ABC^+_{\mathcal{F}_2} = ABCD \neq ABCDF$). We kiezen (zomaar) $ABC{\rightarrow}D$ ter eliminatie. Dus splitsen we $\bar{R}_2$ in $\bar{R}_{2a} = ABCD$ en $\bar{R}_{2b} = ABC\not{D}F = ABCF$. Dit levert schema's $\mathbf{R}_{2j} = (\bar{R}_{2j}, \mathcal{F}_{2j})$, waarbij $\mathcal{F}_{2j}$ (een basis voor) de inperking is van $\mathcal{F}^+$ (of $\mathcal{F}_2^+$) tot $\bar{R}_{2j}$. Dus

$\mathbf{R}_{2a} = (ABCD, \{ABC{\rightarrow}D,\ CD{\rightarrow}B\})$ en
$\mathbf{R}_{2b} = (ABCF, \{\ \})$.

De FDs van $\mathbf{R}_2$ zijn behouden. Schema $\mathbf{R}_{2b}$ staat in BCNF (het heeft geen niet-triviale FDs). In schema $\mathbf{R}_{2a}$ is $CD{\rightarrow}B$ een schending van de BCNF-conditie (want $B \not\subseteq CD$ en $CD^+_{\mathcal{F}_{2a}} = BCD \neq \bar{R}_{2a}$). Decompositie van $\mathbf{R}_{2a}$ levert: $\mathbf{R}_{2a1} = (BCD, \{CD{\rightarrow}B\})$ en $\mathbf{R}_{2a2} = (ACD, \{\ \})$, die beide in BCNF staan en waarbij $ABC{\rightarrow}D$ verloren is gegaan.

- Dus $\{\mathbf{R}_{1a}, \mathbf{R}_{1b}, \mathbf{R}_{2a1}, \mathbf{R}_{2a2}, \mathbf{R}_{2b}\}$ is een decompositie van $\mathbf{R}$ waarvan alle componenten in BCNF staan. Er zijn functionele afhankelijkheden verloren gegaan.

- NB. Omdat dit een lossless decompositie is (een eigenschap van het toegepaste BCNF-algoritme), schrijven we ook wel:

"$ABCDEF = BCE \bowtie ACE \bowtie BCD \bowtie ACD \bowtie ABCF$ geldt in $\mathbf{R}$."

Wanneer je met een andere schending begint kun je mogelijk een andere decompositie bereiken.

In de volgende opgavenserie wordt het volgende databaseschema gebruikt:

Class (name, type, country, guns, bore, displacement)
Ship (name, classname, launched)
Battle (name, date)
Outcome (shipname, battlename, result)

De attributen die tot de sleutel behoren zijn onderstreept.
In *Ship* is *classname* een foreign key verwijzend naar *Class (name)*.
In *Outcome* is *shipname* een foreign key verwijzend naar *Ship (name)*.
In *Outcome* is *battlename* een foreign key verwijzend naar *Battle (name)*.

Schepen die volgens eenzelfde ontwerp worden gebouwd vormen samen een klasse (*class*). Klassen komen in twee typen (*type*): *bb* (voor *battleship*) en *bc* (voor *battlecruiser*). De overige attributen van een klasse zijn: het land (*country*), het aantal kanonnen (*guns*), de diameter in centimeters van de kanonsloop (*bore*), en de waterverplaatsing (*displacement*, gemeten in tonnen). Van een schip is, naast de naam (*name*) en de klassenaam (*classname*), ook nog bekend wanneer het te water is gelaten (*launched*). Van een zeeslag (*battle*) is de naam (*name*) en datum (*date*) bekend. Relatie *Outcome* geeft aan hoe schepen de zeeslagen hebben doorstaan: gezonken, beschadigd of okay (*result* = *sunk*, *damaged*, en *ok*, respectievelijk).

Wanneer we spreken van het *type* van een schip, dan bedoelen we het *type* van de klasse van dat schip; net zo voor de attributen *country*, *guns*, *bore*, *displacement*. Dus alle schepen van een klasse komen uit één land: het land dat in de klasse genoemd staat.

U mag identifiers tot hun eerste letter afkorten. Het schema luidt dan:

C (n, t, c, g, b, d)
S (n, c, l)
B (n, d)
O (s, b, r)

10p. **Opgave 8.** Beschouw de volgende zoekopdracht:

Geef iedere zeeslag waarin schepen van verschillende klassen zijn betrokken.

Geef voor deze vraag een *afleiding in kleine stappen* in verzamelingsnotatie.

De verkregen eindvorm moet dicht aansluiten bij SQL en, na omzetting naar SQL, *geen subqueries hebben en geen overbodige tabellen gebruiken en geen group-by clausules bevatten.* Kort tabel- en attribuutnamen af tot hun eerste letter. Het begin is al gegeven.

Onleesbare tekst wordt fout gerekend.

	"iedere zeeslag waarin schepen van verschillende klassen zijn betrokken"
=	$\{b \mid (\exists\, s, s' \mid$ "klassen van s en s' zijn verschillend" $\bullet$ "s en s' nemen deel aan b") $\bullet\ b.n\}$
=	$\{b \mid (\exists\, s, s' \mid s.c \neq s'.c \bullet (\exists\, o, o' \mid o.s = s.n \wedge o'.s = s'.n \bullet o.b = b.n = o'.b)) \bullet b.n\}$
=	[predicate logic]
	$\{b \mid (\exists\, s, s', o, o' \mid s.c \neq s'.c \bullet o.s = s.n \wedge o'.s = s'.n \wedge o.b = b.n = o'.b) \bullet b.n\}$
=	[shunting]
	$\{b, s, s', o, o' \mid s.c \neq s'.c \wedge o.s = s.n \wedge o'.s = s'.n \wedge o.b = b.n = o'.b \bullet b.n\}$
=	[equals for equals (voorbereiding op de stap erna)]
	$\{b, s, s', o, o' \mid s.c \neq s'.c \wedge o.s = s.n \wedge o'.s = s'.n \wedge o.b = b.n = o'.b \bullet o.b\}$
=	[shunting (voorbereiding op de stap erna)]
	$\{s, s', o, o' \mid s.c \neq s'.c \wedge o.s = s.n \wedge o'.s = s'.n \wedge (\exists\, b \bullet o.b = b.n) \wedge o.b = o'.b \bullet o.b\}$
=	[in O is b een foreign key naar $B(n)$]
	[Dus voor iedere o geldt: $(\exists\, b \bullet o.b = b.n)$]
	$\{s, s', o, o' \mid s.c \neq s'.c \wedge o.s = s.n \wedge o'.s = s'.n \wedge o.b = o'.b \bullet o.b\}$
	We hebben kortheidshalve steeds $s : S$ afgekort tot s, en net zo bij $c{:}C$, $b{:}B$, $o{:}O$.
	(Zie Toelichting.)

Geef een SQL formulering van de beschouwde vraag; de SQL formulering moet dicht aansluiten bij de zojuist gegeven uitdrukking. Gebruik *DISTINCT* alleen wanneer het nodig is.

Onleesbare tekst wordt fout gerekend.

select distinct $o.b$
from S s, S s', O o, O o'
where $s.c{\neq}s'.c$ and $o.s{=}s.n$ and $o'.s{=}s'.n$ and $o.b{=}o'.b$

Geef ook een formulering van de vraag in Relationele Algebra waarin zoveel mogelijk gebruik gemaakt wordt van de natural join ($\bowtie$) in plaats van andere vormen van joins of het Cartesische product.

Onleesbare tekst wordt fout gerekend.

$\pi_{ob}\ (\sigma_{sc' \neq sc''}\ (S' \bowtie O' \bowtie O'' \bowtie S''))$

Hierbij zijn met renaming nieuwe relaties gedefinieerd:

$S' = S[sn', sc', sl']$ en $O' = O[sn', ob, or']$ en

$S'' = S[sn'', sc'', sl'']$ en $O'' = O[sn'', ob, or'']$.

(Let op de gelijkheid en ongelijkheid van de nieuwe attribuutnamen.)

Class (<u>name</u>, type, country, guns, bore, displacement)	*C (<u>n</u>, t, c, g, b, d)*
Ship (<u>name</u>, classname, launched)	*S (<u>n</u>, c, l)*
Battle (<u>name</u>, date)	*B (<u>n</u>, d)*
Outcome (<u>shipname, battlename</u>, result)	*O (<u>s, b</u>, r)*

10p. **Opgave 9.** Formuleer in SQL met een group-by query:

Geef voor ieder type dat in meer dan 10 zeeslagen betrokken is, het aantal schepen van dat type.

We zeggen: "Een schip is van type t" als de klasse van het schip type t heeft. En ook: "Type t is betrokken in een zeeslag b" als er een schip van type t betrokken is in zeeslag b.

Onleesbare tekst wordt fout gerekend.

```
select c1.type, count (distinct s2.name)
from Class c1, Ship s1, Outcome o1, Class c2, Ship s2
    -- c1, s1, o1 is een getuige van het feit dat c1.type betrokken is in een zeeslag
    -- c2, s2 is een getuige van het feit dat er een schip bestaat van c2.type
where c1.type = c2.type
  and c1.name = s1.classname and s1.name = o1.shipname
  and c2.name = s2.classname
group by c1.type
having count (distinct o1.battlename) > 10                    (Zie Toelichting.)
```

5p. **Opgave 10.** (Goede beantwoording levert 5 bonuspunten boven op de punten die voor deze opgave gegeven worden. Daardoor kan het totaal aantal behaalde punten boven de 100 uitkomen.) Beschouw de volgende vraag:

> Van welke zeeslagen zijn alle daarbij betrokken schepen ooit (in een of andere zeeslag) gezonken?

Formuleer de vraag in verzamelingsnotatie op een manier die *zo direct mogelijk* aansluit bij *de gegeven formulering* van de vraag.

Onleesbare tekst wordt fout gerekend.

Kortheidshalve laten we de typering achterwege (en korten dus $b : B$ af tot b, et cetera).

$\{b \mid (\forall\, s \mid$ "s betrokken bij b" $\bullet$ "s ooit gezonken"$) \bullet b.n\}$

$= \{b \mid (\forall\, s \mid (\exists\, o \bullet s.n = o.s \wedge o.b = b.n) \bullet (\exists\, o' \bullet s.n = o'.s \wedge o'.r = \text{sunk})) \bullet b.n\}$

(Zie Toelichting.)

Geef *de formulering in verzamelingsnotatie* van de SQL query die u zo dadelijk gaat geven. (Een afleiding op kladpapier zou u hierbij kunnen helpen!)

Onleesbare tekst wordt fout gerekend.

$\{b \mid \neg\, (\exists\, o \mid o.b = b.n \bullet \neg\, (\exists\, o' \bullet o.s = o'.s \wedge o'.r = \text{sunk})) \bullet b.n\}$

Formuleer de vraag in SQL, zonder overbodige subqueries en tabellen.

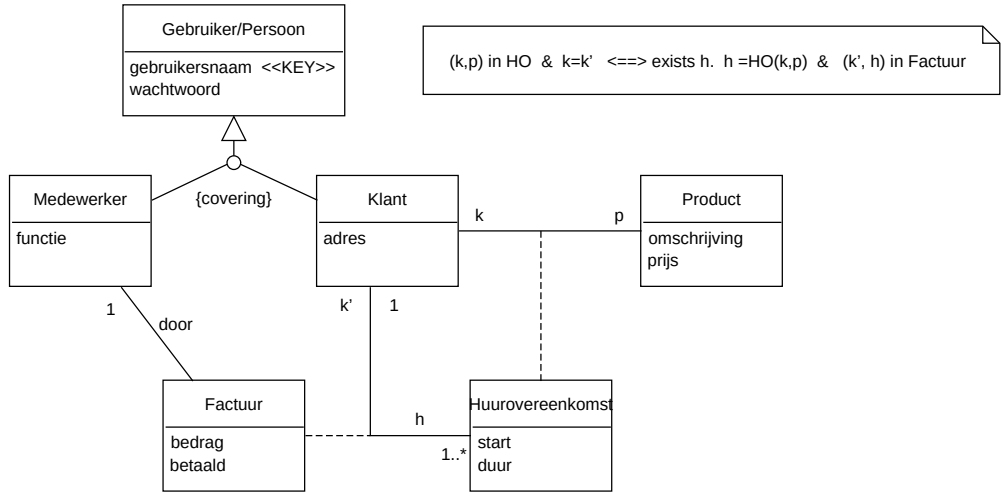
Onleesbare tekst wordt fout gerekend.

```
select b.n from B b where not exists (
    select * from O o where o.b = b.n and not exists (
        select * from O o' where o.s = o'.s and o'.r = sunk))
```

Toelichtingen

Toelichting bij antwoord 1.
(1) Merk op dat per definitie de key van *Huurovereenkomst* bestaat uit een paar (k, p) van klant k en product p. Dus *start* en *duur* van een huurovereenkomst zijn volledig beplaald door een klant en product.
(2) "De twee wegen van *Factuur* naar *Klant* leiden tot dezelfde instantie", dit luidt formeel: $(f, k) \in \mathit{voor} \Leftrightarrow \exists\, p \bullet (f, HO(k, p)) \in \mathit{betreft}$. (Dit is een zogenaamde *cycle-eigenschap*.) Vanwege de equivalentie is relatie *voor* uit te drukken in de andere relaties en kan desgewenst weggelaten worden. Wanneer relatie *voor* niet wordt weggelaten, is de cycle-eigenschap (eigenlijk) nodig!

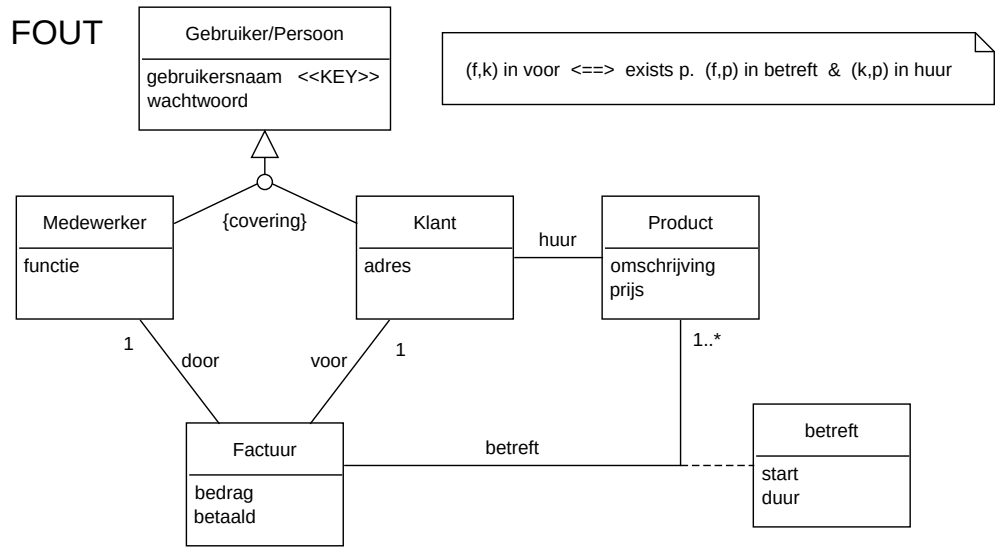
Hier is een alternatieve oplossing:



Zonder de cycle-eigenschap is de oplossing fout.

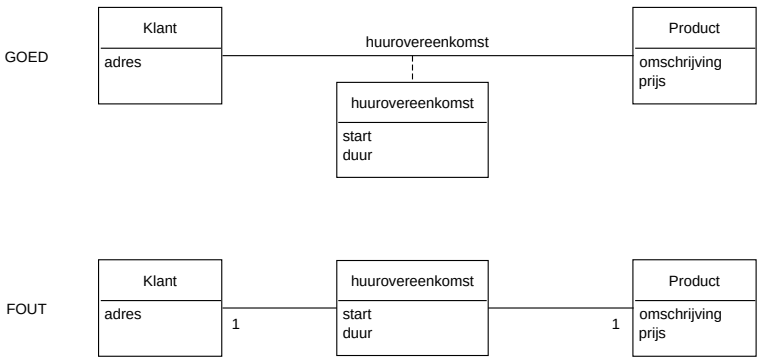
* * *

Hier is een foute oplossing:



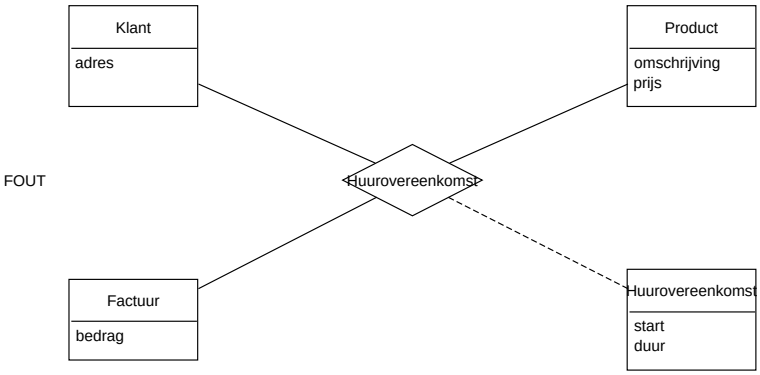
Bij bovenstaand diagram is het (ten onrechte!) mogelijk dat een klant-product combinatie in twee *verschillende* facturen voorkomt; dus zijn *start* en *duur* niet door een klant-product paar uniek bepaald. Deze fout kan verbeterd worden door de eis dat er bij een klant-product combinatie in *huur* hooguit één factuur is die aan hun gerelateerd is via *voor* en *betreft*.

Merk ook het verschil op tussen de volgende twee fragmenten:



In de bovenste is de *Klant-Product* combinatie de key van *Huurovereenkomst*. Dat is in het onderste fragment niet het geval (tenzij het er apart bij ge-eist wordt); in het onderste diagram kan het zo zijn dat er twee verschillende huurovereenkomsten zijn die aan het zelfde tweetal van klant en product zijn gerelateerd.

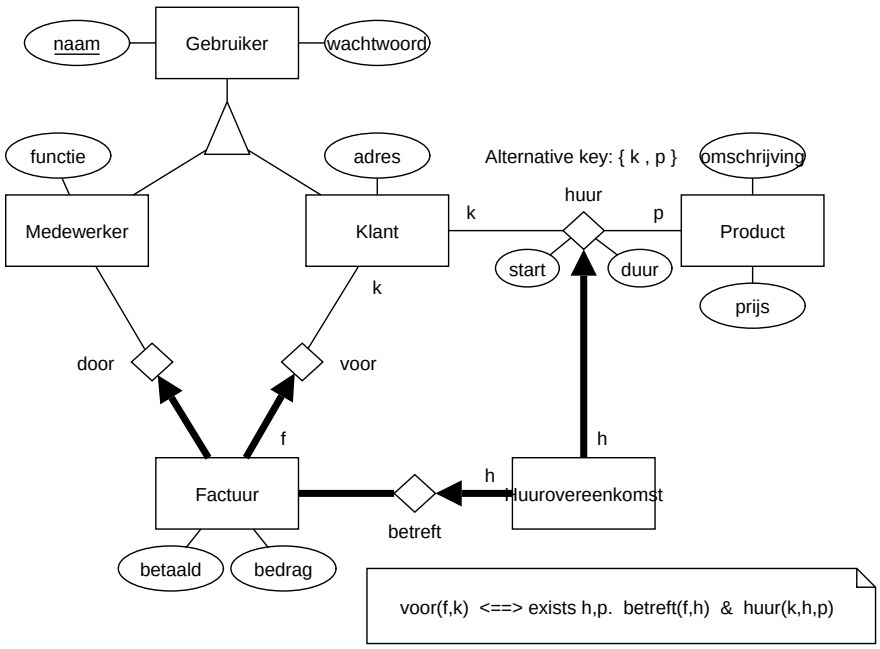
Het is fout om *Huurovereenkomst* tot een relatie te maken tussen méér dan *Klant* en *Product*, zoals hier:



Nu kan, bijvoorbeeld, een combinatie (k, p) van klant en product méérmalen in *Huurovereenkomst* zitten (steeds met een andere factuur).

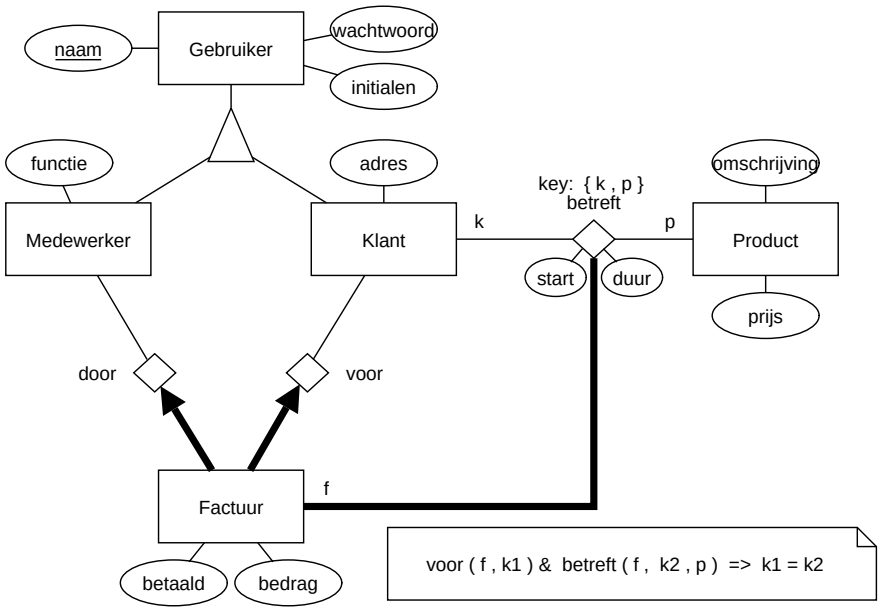
* * *

Hier is de oplossing in de notatie van het boek:



(1) Niet alleen bepaalt een huurovereenkomst zowel de klant als ook het product uniek, zoals aangegeven door de pijl van *Huurovereenkomst* naar *huur* (dat wil zeggen, *Huurovereenkomst* is een sleutel van *huur*), maar ook is het zo dat een klant en product die in de relatie *huur* zitten, samen uniek de huurovereenkomst bepalen: dat wil zeggen, *Klant* en *Product* vormen samen een sleutel van *huur*, zoals aangegeven in door "Alternative key: $\{k, p\}$". Dus *start* en *duur* worden geheel door een klant k en product p bepaald!
(2) "De twee wegen van *Factuur* naar *Klant* leiden tot dezelfde instantie", dit luidt formeel: $(f, k) \in \mathit{voor} \Leftrightarrow \exists\, h, p \bullet (f, h) \in \mathit{betreft} \wedge \mathit{huur}(k, h, p)$. (Dit is een zogenaamde *cycle-eigenschap*.) Vanwege de equivalentie is relatie *voor* uit te drukken in de andere relaties en kan desgewenst weggelaten worden. Wanneer relatie *voor* niet wordt weggelaten, is de cycle-eigenschap (eigenlijk) nodig!
(3) De dikke lijn van *Factuur* naar *betreft* ("*Factuur* participeert in *betreft*") betekent dat iedere factuur deelneemt in de relatie *betreft*, met andere woorden, er zijn geen facturen die niet een huurovereenkomst (dus klant en product) betreffen.
(4) Omdat *Huurovereenkomst* sleutel is van *huur*, is er geen verandering in betekenis wanneer attributen *start* en *duur* bij *Huurovereenkomst* staan of bij *huur*.

Hier is een alternatieve oplossing:



(1) Merk op dat alweer *Klant* en *Product* de sleutel zijn van *betreft* (zoals expliciet vermeld): bij een klant en product is er hoogstens één factuur die met hun in de relatie *betreft* zit. Dus *start* en *duur* worden geheel door *Klant* en *Product* bepaald!
(2) Ook hier geldt dat de twee wegen van *Factuur* naar *Klant* tot dezelfde instantie leiden: als $voor(f, k_1) \wedge betreft(f, k_2, p)$, dan $k_1 = k_2$.
Er geldt zelfs: $voor(f, k) \Leftrightarrow (\exists\, p \bullet betreft(k, f, p))$. Dus relatie *voor* is uit te drukken in de andere relaties en kan desgewenst weggelaten worden.
(3) De dikke lijn van *Factuur* naar *betreft* ("*Factuur* participeert in *betreft*") betekent dat iedere factuur deelneemt in de relatie *betreft*, met andere woorden, er zijn geen facturen die niet een klant en product betreffen.

Toelichting bij antwoord 2.
(1) De check in A representeert "multipliciteit $1 \ldots *$ naar D" van relatie S.
(2) De checks in B en C representeren de *disjoint* eigenschap van de generalisatie.
(3) Middels tabel $C2$ wordt de SET-waardigheid van attribuut $c2$ in C gerepresenteerd. Wanneer tabel $C2$ weggelaten wordt en het schema van C wordt gewijzigd in

$$C(a1, c1, c2, \text{primary key } (a1, c2), \ldots),$$

dan is er redundantie in de opgeslagen waarden ten gevolgde van de functionele afhankelijkheid $a1 \rightarrow c1$.
(4) In $C2$ en S bestaat de primary key uit alle attributen van de tabel en kan weggelaten worden (maar dan moet eigenlijk wel 'not null' voor ieder van die attributen toegevoegd worden).
(5) Een alternatief met één tabel minder (en dus een betere oplossing) is deze: laat tabel A weg en neem zijn attributen op in B en C (dankzij de disjointness van de generalisatie wordt er hierdoor geen redundantie geïntroduceerd) en vervang in S de foreign key naar A door: check ($a1$ in (select $a1$ from B union select $a1$ from C)).

Toelichting bij antwoord 3.
Oorspronkelijk stond er bij (5) "E is de naam van de eigenaar…". Daarmee is $E \rightarrow N$

onwaar, aannemende dat verschillende eigenaren verschillende namen kunnen hebben.

Toelichting bij antwoord 3.
Het is fout om als motivatie te geven: "alle FDs blijven behouden". Beschouw maar eens $\mathcal{R} = (ABCD, \{B{\rightarrow}A,\ C{\rightarrow}D\})$ met $\mathcal{R}_1 = (AB, \{B{\rightarrow}A\})$ en $\mathcal{R}_2 = (CD, \{C{\rightarrow}D\})$. Dan zijn in de decompositie $AB \bowtie CD$ wel alle FDs behouden, maar desondanks $ABCD \neq AB \bowtie CD$.

Toelichting bij antwoord 4.
Het is fout om in *Patient* op te nemen: "foreign key (*patientnr*) references *Factuur*(*patientnr*)", want (i) in *Factuur* is (*patientnr*) geen key (maar slechts een component van een key), en (ii) het zou impliceren dat iedere patiënt tenminste een factuur heeft (en dat is in de casus niet gegeven).

Toelichting bij antwoord 5.
4 punten voor (1), 3 punten voor (2,3), en 3 punten voor (4).
NB 1: Sleutels zijn attribuutverzamelingen; zoiets als "$ABC \rightarrow \ldots$" is geen sleutel (maar misschien wel een functionele afhankelijkheid). Het omcirkelen van de hele functionele afhankelijkheid wordt fout gerekend als alleen de sleutel omcirkeld moet worden. Net zo voor het onderstrepen van de supersleutels.
NB 2: Iedere sleutel is ook een supersleutel; omgekeerd niet. Dus ook sleutels moeten onderstreept worden wanneer gevraagd wordt de supersleutels te onderstrepen.

Toelichting bij antwoord 7.
Oplossing 2. Neem $CD{\rightarrow}B$ ter eliminatie. Dit geeft:
$\mathbf{R}_1 = (BCD, \{CD{\rightarrow}B\})$
$\mathbf{R}_2 = (ACDEF, \{E{\rightarrow}D,\ ACD{\rightarrow}E\})$.
($ACD{\rightarrow}E$ zit niet in $\mathcal{F}$, maar wel in $\mathcal{F}^+$ en bevat alleen maar attributen uit $\bar{R}_2$.) Afhankelijkheid $ABC{\rightarrow}E$ zit wel in $\mathcal{F}$ en niet in $(\mathcal{F}_1 \cup \mathcal{F}_2)^+$ en is dus verloren gegaan.

Oplossing 3. Neem $E{\rightarrow}D$ ter eliminatie. Dit geeft:
$\mathbf{R}_1 = (DE, \{E{\rightarrow}D\})$
$\mathbf{R}_2 = (ABCEF, \{ABC{\rightarrow}E,\ CE{\rightarrow}B\})$.
($CE{\rightarrow}B$ zit niet in $\mathcal{F}$, maar wel in $\mathcal{F}^+$ en bevat alleen maar attr ibuten uit $\bar{R}_2$.) Afhankelijkheid $CD{\rightarrow}E$ zit wel in $\mathcal{F}$ en niet in $(\mathcal{F}_1 \cup \mathcal{F}_2)^+$ en is dus verloren gegaan.

Foute oplossing. We passen niet het BCNF-algoritme toe, maar verzinnen zomaar een decompositie:
$\mathbf{R}_1 = (ABCDE, \{ABC{\rightarrow}E,\ CD{\rightarrow}B,\ E{\rightarrow}D\})$
$\mathbf{R}_2 = (ABCEF, \{\ \})$.
Deze decompositie is verliesvrij (lossless) omdat de intersectie van de attribuutverzamelingen van de schema's een key is in een van beide schema's (ABC is een key in $\mathbf{R}_1$), en er zijn geen FDs verloren gegaan (ze zitten allemaal nog in $\mathbf{R}_1$). Echter, er is geen afname van het aantal BCNF-schendingen: de oorspronkelijke schendingen uit $\mathcal{F}$ zitten nog steeds allemaal in $\mathbf{R}_1$.

Toelichting bij antwoord 8.
Hier is een omweg-afleiding voor het deel "klassen van s en s' zijn verschillend":

"klassen van s en s' zijn verschillend"
$= (\exists\, c, c' \mid c.n = s.c \wedge c'.n = s'.c \bullet c \neq c')$
$=$ [in C is n een key]
$(\exists\, c, c' \mid c.n = s.c \wedge c'.n = s'.c \bullet c.n \neq c'.n)$
$=$ [equals for equals (voorbereiding op de stap erna)]
$(\exists\, c, c' \mid c.n = s.c \wedge c'.n = s'.c \bullet s.c \neq s'.c)$
$=$ [predicate logic]
$(\exists\, c \bullet c.n = s.c) \wedge (\exists\, c' \bullet c'.n = s'.c) \wedge s.c \neq s'.c$
$=$ [in S is c een foreign key naar $C(n)$]
[Dus voor iedere s geldt: $\exists\, c \bullet c.n = s.c$]
$s.c \neq s'.c$

Toelichting bij antwoord 9.
De distinct in de select-expressie is nodig omdateen zelfde combinatie $c2, s2$ bij verschillende keuzen voor $c1, s1, o1$ kan voorkomen.
Klasse $c2$ is nodig (en kan niet door $c1$ vervangen worden) omdat anders alleen schepen geteld worden uit speciale klassen (namelijk klassen die in een zeeslag betrokken zijn).

Alternatieve lelijke oplossing (tellen van de zeeslagen apart binnen de having clausule):
select $c2.type$, count ($s2.name$)
from *Class* $c2$, *Ship* $s2$
where $c2.name = s2.classname$
group by $c2.type$
having 10 < (
  select count (distinct $o1.battlename$) from C $c1$, S $s1$, O $o1$
  where $c1.type = c2.type$ and $c1.name = s1.classname$ and $s1.name = o1.shipname$)

Alternatieve lelijke oplossing (tellen van de schepen apart binnen de select expressie):
select $c1.type$,
    (select count($s2.name$) from C $c2$, S $s2$
     where $c1.type = c2.type$ and $c2.name = s2.classname$)
from *Class* $c1$, *Ship* $s1$, *Outcome* $o1$
where $c1.name = s1.classname$ and $s1.name = o1.shipname$
group by $c1.type$
having count (distinct $o1.battlename$) > 10

Alternatieve lelijke oplossing (tellingen van de zeeslagen en van de schepen beide apart):
select $c.type$,
    (select count($s2.name$) from C $c2$, S $s2$
     where $c.type = c2.type$ and $c2.name = s2.classname$)
from *Class* c
group by $c.type$
having 10 < (
  select count (distinct $o1.battlename$) from C $c1$, S $s1$, O $o1$
  where $c.type = c1.type$ and $c1.name = s1.classname$ and $s1.name = o1.shipname$)

Toelichting bij antwoord 10.
De afleiding gaat verder als volgt:

$$
\begin{array}{ll}
 & \ldots \\
= & \quad [\text{shunting}] \\
 & \{b \mid (\forall\, s, o \mid s.n = o.s \wedge o.b = b.n \bullet (\exists\, o' \bullet s.n = o'.s \wedge o'.r = \text{sunk})) \bullet b.n\} \\
= & \quad [\text{equals for equals (voorbereiding op de stap erna)}] \\
 & \{b \mid (\forall\, s, o \mid s.n = o.s \wedge o.b = b.n \bullet (\exists\, o' \bullet o.s = o'.s \wedge o'.r = \text{sunk})) \bullet b.n\} \\
= & \quad [\text{shunting (voorbereiding op de stap erna)}] \\
 & \{b \mid (\forall\, o \mid (\exists\, s \bullet s.n = o.s) \wedge o.b = b.n \bullet (\exists\, o' \bullet o.s = o'.s \wedge o'.r = \text{sunk})) \bullet b.n\} \\
= & \quad [\text{in } O \text{ is } s \text{ een foreign key naar } S(n)] \\
 & \quad [\text{Dus voor iedere } o \text{ geldt: } (\exists\, s \bullet s.n = o.s)] \\
 & \{b \mid (\forall\, o \mid o.b = b.n \bullet (\exists\, o' \bullet o.s = o'.s \wedge o'.r = \text{sunk})) \bullet b.n\} \\
= & \{b \mid \neg\, (\exists\, o \mid o.b = b.n \bullet \neg\, (\exists\, o' \bullet o.s = o'.s \wedge o'.r = \text{sunk})) \bullet b.n\}
\end{array}
$$