Tentamen Gegevensbanken (19.211074.1) — 1 februari 2013

CONTROLEER EERST OF ALLE BLADZIJDEN T/M BLZ. 16 AANWEZIG 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.

[Reeds op Blackboard aangekondigd:] Bij het tentamen mogen geen boeken en dergelijke gebruikt worden behoudens **één dubbelzijdig gebruikt** vel van A4-formaat met daarop eigen aantekeningen of kopieën van delen van het boek; **kopieën van tentamenuitwerkingen en ander materiaal zijn niet toegestaan** (dat moet dan maar in eigen aantekeningen verwerkt worden).

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	4	5	6	7	8	bonus	9	10

10p. **Opgave 1.** De organisatie *Scientific Lectures* organiseert lezingen voor een breed publiek. De lezingen kunnen zowel *besloten* gegeven worden als ook *openbaar*; in het eerste geval (besloten lezing) is er een bedrijf dat voor de lezing betaalt en kunnen alle bedrijfsmedewerkers de lezing bijwonen, terwijl in het tweede geval (openbare lezing) moeten personen individueel voor de lezing betalen. Hier is een beschrijving van een enigszins vereenvoudigde werkelijkheid:

- Voor iedere lezing is precies één staflid van *Scientific Lectures* verantwoordelijk.
- De organisatie *Scientific Lectures* kent een collectie potentiële sprekers. Iedere potentiële spreker die daadwerkelijk een lezing geeft, krijgt daarvoor een beloning — precies één beloning per lezing. Bij *Scientific Lectures* is bekend wie welke beloning krijgt voor welke lezing. Per lezing is er minstens één spreker.
- Een lezing is besloten of openbaar, maar niet beide tegelijk. Bij een besloten lezing is bekend welk bedrijf voor de lezing betaalt. Bij een openbare lezing is de toegangsprijs en locatie bekend.
- Er is een collectie potentiële toehoorders die de openbare lezingen kunnen bijwonen. Voor iedere potentiële toehoorder die daadwerkelijk bij een lezing aanwezig is, is bekend welk staflid van *Scientific Lectures* het toegangsgeld voor die lezing heeft ontvangen.
- Van een potentiële spreker is NAW (Naam, Adres en Woonplaats) bekend. Van een toehoorder is een e-mailadres bekend. Van een staflid is een medewerkersnummer bekend; verschillende stafleden hebben verschillende nummers.

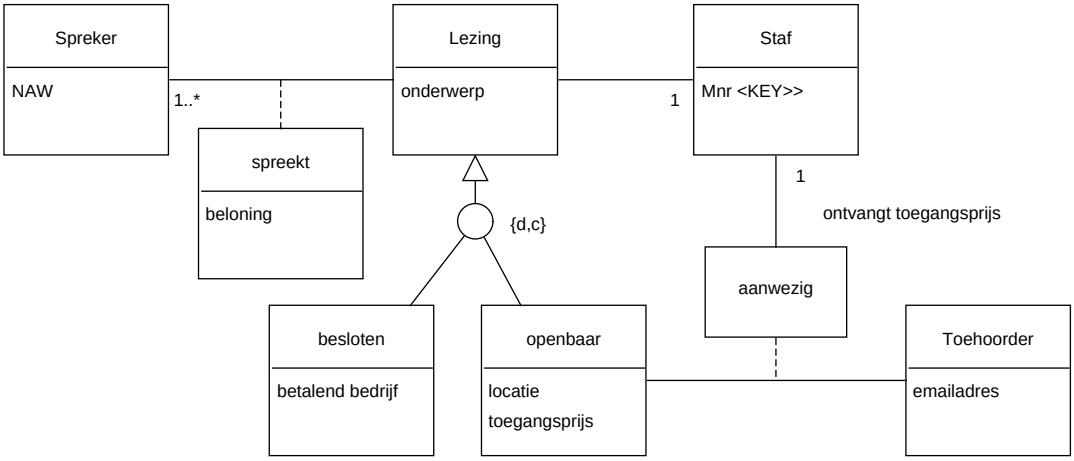
Geef in het antwoordblok een Entity-Relationship diagram dat zoveel als mogelijk de volgende eigenschappen heeft:

- iedere instantie van het ERD beschrijft een mogelijke "werkelijkheid *op één tijdstip*",
- iedere mogelijke "werkelijkheid *op één tijdstip*" kan gerepresenteerd worden als een instantie van het ERD,
- het ERD is opgebouwd met *zo geschikt mogelijke* constructies.

Zowel de ERD-notatie uit het boek als ook de UML notatie (class diagram) is toegestaan, *maar een mengeling van beide niet.* *Doe het eerst op kladpapier en dan pas in het net.*

Onleesbare tekst wordt fout gerekend.

(Niet-vermelde multipliciteiten leggen geen beperkingen op: $0 \ldots *$.)



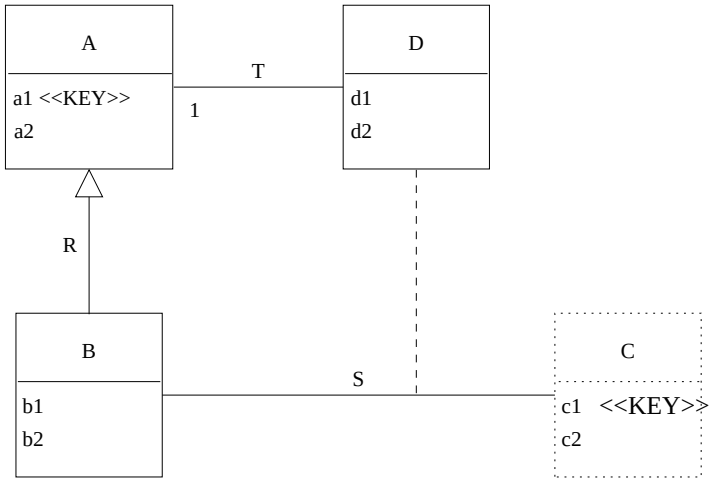
(Zie Toelichting.)

Hier is ruimte voor eventuele toelichting en verklaring van verschillen tussen casus en ERD:

Onleesbare tekst wordt fout gerekend.

De aanduiding $\{d, c\}$ bij de specialisatieknopp staat voor *disjunct* en *covering*; iedere lezing is besloten of openbaar en niet beide tegelijk.

10p. **Opgave 2.** Beschouw het volgende ER-diagram in de notatie van de UML (niet-geschreven multipliciteiten staan voor $0 \,..\, *$):



De stippellijn waarmee entiteittype C getekend is, betekent dat het niet van belang is welke instanties er van C bestaan, en het entiteittype alleen maar nodig is om de rest van het ERD (met name relatie S) te kunnen formuleren.

Vertaal het ER-diagram naar een databaseschema dat *precies* de informatie op kan slaan die past in het ER-diagram, *behalve* de informatie die zegt "welke instanties van C bestaan" (er hoeft dus geen tabel C te komen), en voldoet aan de volgende eigenschappen:

- er zijn geen NULLs nodig vanwege de vertaling,
- er wordt geen redundantie geïntroduceerd door de vertaling,
- alle attributen hebben atomaire waarden,

en verder, zoveel als mogelijk na vervulling van voorgaande eisen,

- er zijn *zo weinig mogelijk* tabelschema's.

Geef de tabelschema's in SQL syntaxis waarbij domeinen weggelaten mogen worden; bijv:
$X(x_1, \ldots,$ primary key $(x_i, x_j ...),$ foreign key $(x_m, x_n ...)$ references $Y(y_1, y_2 ...), \ldots)$
Gebruik afkortingen, zoals "PK" voor "primary key", etc., en gebruik zo nodig ook CHECKs.

Onleesbare tekst wordt fout gerekend.

$A(a1, a2,$ PK $a1);$

$B(a1, b1, b2,$ PK $(a1),$ FK $(a1)$ REF $B(a1));$

$D(a1B, c1, c2, d1, d2, a1A,$ -- D en S zijn hetzelfde

PK $(a1B, c1),$ -- de key van S (en dus D) is per definitie: (key van B, key van C)

FK $(a1B)$ REF $B(a1),$

-- FK $(c1)$ REF $C(c1),$ ← niet doen omdat er geen tabel C is

FK $(a1A)$ REF $A(a1),$ CHECK ($a1A$ IS NOT NULL) ← modelleert relatie T

);

-- $C(c1, c2,$ PK$(c1))$ ← niet doen; er wordt gevraagd C niet te modelleren

Afwezigheid van de 'not null' constraint wordt niet fout gerekend.

(Zie Toelichting.)

10p. **Opgave 3.** Beschouw het relatieschema $\mathbf{R} = (R, \mathcal{F})$, waarbij de attribuutverzameling R en de verzameling $\mathcal{F}$ van functionele afhankelijkheden als volgt luiden:

$$\begin{aligned} R &= ABCDE \\ \mathcal{F} &= \{A \to B, \quad B \to A, \quad BC \to D, \quad DE \to A\} \end{aligned}$$

(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 $X^+_{\mathcal{F}}$). U hoeft de leden van X niet op te nemen in Y.
(2) *Omcirkel* in het antwoordblok de *sleutels* van $\mathbf{R}$.
(3) *Onderstreep* in het antwoordblok de *supersleutels* van $\mathbf{R}$.
(4) Omcirkel in het antwoordblok de *nummers* van de functionele afhankelijkheden die een schending vormen van de BCNF-eis.

Onleesbare tekst wordt fout gerekend.

	X	$\to$	Y
1		$\to$	
2	A	$\to$	B
3	B	$\to$	A
8	AC	$\to$	BD
9	AD	$\to$	B
10	AE	$\to$	B
11	BC	$\to$	AD
12	BD	$\to$	A
13	BE	$\to$	A
14	CD	$\to$	
15	CE	$\to$	
16	DE	$\to$	AB
17	ABC	$\to$	D
18	ABD	$\to$	
19	ABE	$\to$	
20	ACD	$\to$	B
21	ACE	$\to$	BD
22	ADE	$\to$	B
23	BCD	$\to$	A

(Zie Toelichting.)

10p. **Opgave 4.** Beschouw het relatieschema $\mathbf{R} = (ABCDEFG, \mathcal{F})$, waarbij:

$$\mathcal{F} \;=\; \{AB{\rightarrow}CD, \quad D{\rightarrow}FG, \quad E{\rightarrow}D\}$$

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

Onleesbare tekst wordt fout gerekend.

Omdat AB en E niet in het rechterlid van een FD voorkomen, moeten zij tot elke sleutel behoren, en omdat bovendien $ABE^+_{\mathcal{F}} = ABCDEFG$, is ABE de enige sleutel. Van de gegeven $\mathcal{F}$ zijn alle leden een schending van de BCNF-eis voor $\mathbf{R}$; bijvoorbeeld voor $AB{\rightarrow}CD$: de FD is niet-triviaal en het linkerlid omvat geen sleutel (dat wil zeggen, het linkerlid is geen supersleutel).

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 $\mathbf{R}$. *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.

We passen een stap van het BCNF-algoritme toe.

We kiezen (zomaar) de eerste schending, $AB{\rightarrow}CD$, ter eliminatie.

Dus splitsen we $\bar{R}$ in $\bar{R}_1 = ABCD$ en $\bar{R}_2 = AB\not{C}\not{D}EFG = ABEFG$.

Dit levert schema's $\mathbf{R}_i = (\bar{R}_i, \mathcal{F}_i)$, waarbij $\mathcal{F}_i$ gelijk is aan (of equivalent met) de inperking van $\mathcal{F}^+$ tot $\bar{R}_i$.

Dus, $\mathbf{R}_1 = (ABCD, \{AB{\rightarrow}CD\})$ en $\mathbf{R}_2 = (ABEFG, \{AB{\rightarrow}FG,\ E{\rightarrow}FG\})$.

Let op: de FD $AB{\rightarrow}FG$ zit weliswaar niet in $\mathcal{F}$, maar de FD zit wel in $\mathcal{F}^+$ en dus, omdat al zijn attributen in $\bar{R}_2$ zitten, ook in $\mathcal{F}_2$. Net zo voor $E{\rightarrow}FG$.

NB. De decompositie is verliesvrij want $R_1 \cap R_2, {=}AB$, is een key van van $\mathbf{R}_1$ of $\mathbf{R}_2$ (namelijk van $\mathbf{R}_1$).

(Een geheel andere decompositie wordt verkregen door de schending $D{\rightarrow}FG$ of $E{\rightarrow}D$ als uitgangspunt te kiezen. Zie Toelichting bij het laatste onderdeel.)

Zijn in de zojuist gegeven decompositie alle functionele afhankelijkheden behouden? Zo nee, geef dan een afhankelijkheid die niet is behouden:

Onleesbare tekst wordt fout gerekend.

De afhankelijkheid $D{\rightarrow}FG$ is verloren gegaan: $D{\rightarrow}FG \notin (\mathcal{F}_1 \cup \mathcal{F}_2)^+$. Net zo voor $E{\rightarrow}D$.

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 kort maar zódanig dat het voor de corrector duidelijk is hoe u te werk gaat.

Onleesbare tekst wordt fout gerekend.

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

We passen het BCNF-algoritme toe.

- De eerste stap is in de vorige deelvraag gedaan en levert de volgende decompositie:
$\mathbf{R}_1 = (ABCD, \{AB \rightarrow CD\})$ en $\mathbf{R}_2 = (ABEFG, \{AB \rightarrow FG,\ E \rightarrow FG\})$.

- We bekijken nu $\mathbf{R}_1 = (ABCD, \{AB \rightarrow CD\})$.
In $\mathbf{R}_1$ is AB een sleutel, dus is $AB \rightarrow D$ geen schending van de BCNF-conditie, en dus staat $\mathbf{R}_1$ in BCNF.

- We bekijken nu $\mathbf{R}_2 = (ABEFG, \{AB \rightarrow FG,\ E \rightarrow FG\})$.
In $\mathbf{R}_2$ zijn $AB \rightarrow FG$ en $E \rightarrow FG$ beide een schending van de BCNF-conditie; voor $AB \rightarrow FG$ is de reden dat die niet-triviaal is en AB geen supersleutel is in $\mathbf{R}_2$ (want $AB^+_{\mathcal{F}_2} = ABFG \neq ABEFG$). We kiezen (zomaar) $AB \rightarrow FG$ ter eliminatie. Dus splitsen we $\bar{R}_2$ in $\bar{R}_{2a} = ABFG$ en $\bar{R}_{2b} = ABEF\not{G} = ABE$. Dit levert schema's $\mathbf{R}_{2j} = (\bar{R}_{2j}, \mathcal{F}_{2j})$, waarbij $\mathcal{F}_{2j}$ gelijk is aan (of equivalent met) de inperking is van $\mathcal{F}^+$ (of $\mathcal{F}_2^+$) tot $\bar{R}_{2j}$. Dus $\mathbf{R}_{2a} = (ABFG, \{AB \rightarrow FG\})$ (let op: de $AB \rightarrow FG$ zit in $\mathcal{F}_2^+$ en dus in $\mathcal{F}_{2a}$) en $\mathbf{R}_{2b} = (ABE, \{\ \})$ (let op: de FD $E \rightarrow FG$ is verloren gegaan.)

- We bekijken nu $\mathbf{R}_{2a} = (ABFG, \{AB \rightarrow FG\})$. Deze staat in BCNF.

- We bekijken nu $\mathbf{R}_{2b} = (ABE, \{\ \})$. Deze staat in BCNF.

- Dus $\{\mathbf{R}_1, \mathbf{R}_{2a}, \mathbf{R}_{2b}\}$ is een decompositie van $\mathbf{R}$ waarvan alle componenten in BCNF staan. Er zijn functionele afhankelijkheden verloren gegaan; met name zijn dat $D \rightarrow FG$ en $E \rightarrow D$ en $E \rightarrow FG$.

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

  "$ABCDEFG = ABCD \bowtie ABFG \bowtie ABE$ geldt in $\mathbf{R}$."

Wanneer je met een andere schending begint kun je mogelijk een andere decompositie bereiken, en een andere uitslag voor het behoud van de functionele afhankelijkheden. *(Zie Toelichting.)*

10p. **Opgave 5.** Bij een wielerwedstrijd spelen verscheidene entiteiten een rol: ploeg, renner, sponsor, volgauto, etcetera. We definiëren een relatie $\mathcal{R}$ die deze entiteiten aan elkaar relateert, op de volgende manier.
Een tuple (P, R, W, S, K, B) zit op tijdstip t in relatie $\mathcal{R}$ precies wanneer op tijdstip t al het volgende geldt:

1. P is de Ploegnaam van een ploeg die aan de wedstrijd deelneemt.
2. R is het Rugnummer van een wielrenner uit de ploeg met naam P.
3. W is de naam van de Wielrenner met rugnummer R.
4. S is een Sponsor van de ploeg met naam P.
5. K is het Kenteken van een volgauto van de ploeg met naam P.
6. B is het Bedrag dat sponsor S aan wielrenner R geeft in deze wedstrijd.

Neem het volgende aan:

- a. Verschillende ploegen hebben verschillende ploegnamen.
- b. Een rugnummer identificeert een wielrenner uniek.
- c. En ploeg kan verschillende sponsoren hebben.

Geef voor ieder van de functionele afhankelijkheden (FD) en multi-valued dependencies (MVD) in het antwoordblok, met een letter W of O aan of die Waar of Onwaar is in relatie $\mathcal{R}$, en motiveer uw keuze beknopt, zo mogelijk met verwijzing naar 1. .6 en $a \,..\, c$. (De motivatie telt even zwaar mee in de beoordeling als het antwoord W/O.)

FD	W/O	Motivatie voor uw keuze
$R \to P$	W	Volgt uit 2&b
$P \to R$	O	Een ploeg kan verscheidene wielrenners (dus rugnummers) hebben.
$W \to P$	O	Verschillende renners (uit verschillende ploegen) kunnen gelijknamig zijn.
$K \to P$	W	Volgt uit 5 (**aangenomen dat** verschillende ploegen verschillende volgauto's hebben — dit is niet duidelijk uit de casus).
$P \to K$	O	Een ploeg kan verscheidene volgauto's hebben.
$SR \to B$	W	Volgt uit 6 ("het" bedrag).
$BS \to R$	O	Een sponsor kan aan verschillende renners eenzelfde bedrag geven.
$B \to SR$	O	Er geldt al $B \not\to R$: Verschillende renners kunnen eenzelfde bedrag krijgen. Er geldt al $B \not\to S$: Verschillende sponsoren kunnen eenzelfde bedrag geven.
$RB \to S$	O	Verschillende sponsoren kunnen een renner eenzelfde bedrag geven.
$PRWSKB = PRWSK \bowtie SRB$	W	doorsnee der operanden ($=SR$) omvat (zelfs: is) key in rechter operand; SR is key vanwege $SR \to B$
$PRWSKB = PRWSK \bowtie PSRB$	W	doorsnee der operanden ($=PSR$) omvat key SR van rechter operand; SR is key vanwege $SR \to B$ en $R \to P$
$PRWSKB = PRWSK \bowtie SB$	O	Neem een legale instantie van $\mathcal{R}$ met rijen $(...r_1...s...b_1)$ en $(...r_2...s...b_2)$ en $b_1 \neq b_2$. Decompositie en daarna joinen, geeft de rij $(...r_1...s...b_2)$; die zit vanwege de geldige $SR \to B$ niet in de oorspronkelijke instantie.

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 kanonloop (*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)

Samengevat:

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

10p. **Opgave 6.** Beschouw de volgende zoekopdracht:

Vind ieder tweetal van klasse c en zeeslag b waarvoor er (tenminste) twee verschillende schepen bestaan die van klasse c zijn en deelnemen aan zeeslag b.

Geef voor deze vraag een afleiding in kleine stappen in verzamelingsnotatie naar een vorm die dicht aansluit bij een SQL query die geen subqueries heeft en zo weinig mogelijk tabellen in de from clause. Het begin is al gegeven; dat moet je gebruiken. Kort identifiers tot hun eerste letter af, en schrijf desgewenst c voor $c : C$, etc.

Onleesbare tekst wordt fout gerekend.

	"alle paren c; b zó dat: er bestaan schepen $s_1 \neq s_2$ waarbij elke s_i van c is en deelneemt aan b
=	$\{c;\ b \mid (\exists\, s_1, s_2 \mid s_1 \neq s_2 \bullet$ "s_1 en s_2 zijn van klasse c en nemen deel aan b") $\bullet\ (c.n, b.n)\}$
=	$\{c;\ b \mid (\exists\, s_1, s_2 \mid s_1 \neq s_2 \bullet s_1.c = c.n = s_2.c \wedge$ "s_1 en s_2 nemen deel aan b") $\bullet\ (c.n, b.n)\}$
=	$\{c;\ b \mid (\exists\, s_1, s_2 \mid s_1 \neq s_2 \bullet s_1.c = c.n = s_2.c \wedge$
	$(\exists\, o_1, o_2 \bullet s_1.n = o_1.s \wedge s_2.n = o_2.s \wedge o_1.b = b.n = o_2.b)) \bullet (c.n, b.n)\}$
=	[predicatenlogica: "$\exists\,\exists = \exists$"]
	$\{c;\ b \mid (\exists\, s_1, s_2;\ o_1, o_2 \mid s_1 \neq s_2 \bullet s_1.c = c.n = s_2.c \wedge$
	$s_1.n = o_1.s \wedge s_2.n = o_2.s \wedge o_1.b = b.n = o_2.b) \bullet (c.n, b.n)\}$
=	[shunting]
	$\{c;\ b;\ s_1, s_2;\ o_1, o_2 \mid s_1 \neq s_2 \wedge s_1.c = c.n = s_2.c \wedge$
	$s_1.n = o_1.s \wedge s_2.n = o_2.s \wedge o_1.b = b.n = o_2.b \bullet (c.n, b.n)\}$
=	["equals for equals", etc, als voorbereiding voor eliminatie van b en c verderop]
	$\{c;\ b;\ s_1, s_2;\ o_1, o_2 \mid s_1 \neq s_2 \wedge s_1.c = c.n \wedge s_1.c = s_2.c \wedge$
	$s_1.n = o_1.s \wedge s_2.n = o_2.s \wedge o_1.b = b.n \wedge o_1.b = o_2.b \bullet (s_1.c, o_1.b)\}$
=	[shunting (tweemaal) — met geschikte keuze voor de claims van de nieuwe $(\exists \ldots)$]
	$\{s_1, s_2;\ o_1, o_2 \mid s_1 \neq s_2 \wedge (\exists\, c \bullet s_1.c = c.n) \wedge s_1.c = s_2.c \wedge$
	$s_1.n = o_1.s \wedge s_2.n = o_2.s \wedge (\exists\, b \bullet o_1.b = b.n) \wedge o_2.b = o_1.b \bullet (s_1.c, o_1.b)\}$
=	[in S is c een foreign key naar $C(n)$, dus voor iedere s geldt: $\exists\, c \bullet s.c = c.n$]
	[in O is b een foreign key naar $B(n)$, dus voor iedere o geldt: $\exists\, b \bullet o.b = b.n$]
	[propositielogica: constraint $true$ kan weggelaten worden]
	$\{s_1, s_2;\ o_1, o_2 \mid s_1 \neq s_2 \wedge s_1.c = s_2.c \wedge s_1.n = o_1.s \wedge s_2.n = o_2.s \wedge o_2.b = o_1.b \bullet (s_1.c, o_1.b)\}$
=	[in S is n een key, dus voor iedere s, s' geldt: $s{=}s' \Leftrightarrow s.n{=}s'.n$]
	$\{s_1, s_2;\ o_1, o_2 \mid s_1.n \neq s_2.n \wedge$
	$s_1.c = s_2.c \wedge s_1.n = o_1.s \wedge s_2.n = o_2.s \wedge o_2.b = o_1.b \bullet (s_1.c, o_1.b)\}$

Geef een SQL formulering van de beschouwde vraag; de SQL formulering moet dicht aansluiten bij de zojuist gegeven uitdrukking (en dus geen subqueries hebben). Gebruik `DISTINCT` alleen wanneer het nodig is.

Onleesbare tekst wordt fout gerekend.

select distinct $s1.c$, $o1.b$

from S $s1$, S $s2$, O $o1$, O $o2$

where $s1.n \neq s2.n$ and $s_1.n = o_1.s$ and $s_2.n = o_2.s$ and $o_2.b = o_1.b$

NB: 'distinct' is nodig omdat een tweetal (klasse, zeeslag) anders meermalen kan voorkomen.

Vertaal de volgende query naar Relationele Algebra, zonder het Cartesisch product $\times$ te gebruiken maar in plaats daarvan alleen de natural join $\bowtie$ (die koppelt rijen precies wanneer gelijknamige attributen gelijke waarden hebben):

select distinct $c.n$ from C c, S s, O o where $s.c = c.n$ and $o.s = s.n$ and $o.r =$ sunk

Onleesbare tekst wordt fout gerekend.

$\pi_{cn}(\sigma_{or=\text{sunk}}(C[cn,...] \bowtie S[sn,cn,sl] \bowtie O[sn,ob,or])) \quad =$

$\pi_{cn}(\sigma_{or=\text{sunk}}(C' \bowtie S' \bowtie O')) \qquad = \qquad \pi_{cn}(C' \bowtie S' \bowtie \sigma_{or=\text{sunk}}(O'))$

waarbij $C' = C[cn,...]$, $S' = S[sn,cn,sl]$, en $O' = O[sn,ob,or]$ (renaming).

10p. **Opgave 7.** Formuleer in SQL met een group-by query:

Geef ieder paar van land en zeeslag waarbij er minstens twee schepen van dat land betrokken zijn in die zeeslag.
Geef bij zo'n land-zeeslag paar óók het aantal van de schepen van dat land die betrokken zijn bij die zeeslag.

Onleesbare tekst wordt fout gerekend.

select $c.c$, $o.b$, count $(s.n)$

from C c, S s, O o

where $c.n = s.c$ and $s.n = o.s$

group by $c.c$, $o.b$

having count $(s.n) \geq 2$

Vervanging van het deel 'count $(s.n)$' door 'count (distinct $s.n$)' of 'count $(*)$' wijzigt de uitkomst niet. *(Zie Toelichting.)*

5p. **Opgave 8.** (Goede beantwoording levert 5 bonuspunten boven op de 5 punten die voor deze opgave gegeven worden. Daardoor kan het totaal aantal behaalde punten op 105 uitkomen.) Beschouw de volgende vraag:

> Geef de schepen s waarvoor geldt dat voor iedere zeeslag b waaraan s deelneemt, er een schip bestaat dat verschilt van s en óók deelneemt aan b.

Formuleer de vraag geheel in verzamelingsnotatie op een manier die zo direct mogelijk aansluit bij de gegeven formulering van de vraag (kort $s : S$ af tot s, et cetera):

Onleesbare tekst wordt fout gerekend.

(Alleen de laatste regel wordt gevraagd)

$\{s \mid (\forall\, b \mid$ "s neemt deel aan b" $\bullet\ (\exists\, s' \bullet s' \neq s \wedge$ "s' neemt deel aan b")) $\bullet\ s\}$

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

Formuleer de vraag in verzamelingsnotatie op een manier die zo direct mogelijk aansluit bij de SQL query die u zo dadelijk gaat geven:

Onleesbare tekst wordt fout gerekend.

(Alleen de één-na-laatste regel wordt gevraagd)

$= \{s \mid (\forall\, b; o \mid s.n = o.s \wedge o.b = b.n \bullet (\exists\, s'; o' \bullet s'.n \neq s.n \wedge s'.n = o'.s \wedge o'.b = b.n)) \bullet s\}$

$\overset{1}{=} \{s \mid (\forall\, b; o \mid s.n = o.s \wedge o.b = b.n \bullet (\exists\, s'; o' \bullet o'.s \neq s.n \wedge s'.n = o'.s \wedge o'.b = o.b)) \bullet s\}$

$\overset{2}{=} \{s \mid (\forall\, o \mid s.n{=}o.s \wedge (\exists\, b \bullet o.b{=}b.n) \bullet (\exists\, o' \bullet o'.s{\neq}s.n \wedge (\exists\, s' \bullet s'.n{=}o'.s) \wedge o'.b{=}o.b)) \bullet s\}$

$\overset{3}{=} \{s \mid (\forall\, o \mid s.n = o.s \bullet (\exists\, o' \bullet o'.s \neq s.n \wedge o'.b = o.b)) \bullet s\}$

$\overset{4}{=} \{s \mid \neg\, (\exists\, o \mid s.n = o.s \bullet \neg\, (\exists\, o' \bullet o'.s \neq s.n \wedge o'.b = o.b)) \bullet s\}$

1: equals for equals, 2: shunting, 3: in O zijn b/s FK naar $B(n)/S(n)$, 4: "$\forall = \neg\, \exists \neg$"

Formuleer de vraag in SQL, *zonder overbodige subqueries en tabellen*:

Onleesbare tekst wordt fout gerekend.

$=$ select $s.n$ from S s where not exists (

select $*$ from O o where $s.n = o.s$ and not exists (

select $*$ from O $o1$ where $o1.s <> s.n$ and $o1.b = o.b$))

10p. **Opgave 9.** Een bibliotheek wil een database opzetten volgens de volgende richtlijnen:

- Er is een tabel $Titel$ met attributen $isbn$, $titeltekst$, $auteur$. Attribuut $isbn$ is de primary key van $Titel$.
- Er is een tabel $Boek$ met attributen $isbn$, $volgnummer$, $gewicht$, $kast$. De primary key van $Boek$ is het paar $(isbn, volgnummer)$. Een boek *hoort bij* de titel die hetzelfde $isbn$ heeft als het boek.

(Een $Titel$ representeert dus niet één fysiek object, een $Boek$ wel.)
Bij iedere wijziging van de database moet het volgende gelden:

- Een boek B kan niet aan de database toegevoegd worden wanneer er geen titel in de database bestaat met hetzelfde $isbn$ als B.
- Wanneer van een titel in de database de $isbn$-waarde wijzigt, moet dat bij alle boeken die bij die titel horen, ook eensgelijks gebeuren (zodat die boeken bij dezelfde titel blijven horen).
- Een verwijdering van een titel moet tot gevolg hebben dat alle boeken die bij die titel horen, ook verwijderd worden.

Geef de SQL create statements voor $Titel$ en $Boek$ zodanig dat bovenstaande eisen vervuld zijn. U mag géén triggers gebruiken, en u hoeft de domeinen van attribuutwaarden (zoals CHAR(20) en INTEGER) niet te vermelden.
Ter herinnering, foreign key constraints zien er als volgt uit:

foreign key $(x, x', ...)$ references $T(y, y', ...)$ on $event_1$ $action_1$ on $event_2$ $action_2$ ...

Hierbij staat x voor een attribuutnaam, T voor een tabelnaam, $event_i$ voor een gebeurtenis ($delete$, $update$) en $action_i$ voor een actie ($set\ null$, $set\ default$, $no\ action$, $cascade$).

Onleesbare tekst wordt fout gerekend.

create table $Titel$ (

$isbn$, $titeltekst$, $auteur$, primary key ($isbn$))

create table $Boek$ (

$isbn$, $volgnummer$, $gewicht$, $kast$,

primary key ($isbn$, $volgnummer$),

foreign key ($isbn$) references $Titel(isbn)$

on delete cascade on update cascade)

NB. Voor de eerste eis is nodig dat in $Boek$ geldt: $isbn$ is not null; daaraan is voldaan doordat $isbn$ een onderdeel is van de primary key van $Boek$.

De eisen voor databasewijzigingen kunnen ook met triggers vervuld worden in plaats van constraints in de create table statements. Geef de SQL create statements voor $Titel$ en $Boek$ en voor een geschikte trigger zodanig dat *de laatste van* bovenstaande eisen vervuld wordt door de trigger.
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-to-refer-to-old-tuple*]
[new as *var-to-refer-to-new-tuple*]
[old table as *name-to-refer-to-old-table*]
[new table as *name-to-refer-to-new-table*]]
[for each { row | statement }]
[when (*precondition*)]
statement-list

Hierbij staat $\{x \mid y \mid \dots\}$ voor een keuze uit $x, y \dots$; en $[x]$ staat voor een keuze uit x of niets.

Onleesbare tekst wordt fout gerekend.

create table $Titel$ -- als voorheen
create table $Boek$ -- als voorheen maar zonder "on delete cascade"
create trigger $OnDeleteCascade$
after delete on $Titel$
referencing old as O
for each row
delete from $Boek$ B where $B.isbn = O.isbn$

10p. **Opgave 10.** Leg uit wat *Atomicity* betekent in de context van transacties.

Onleesbare tekst wordt fout gerekend.

Atomicity betekent dat van iedere transactie de uitvoering ervan *geheel* of *helemaal niet* plaats vindt, voor zover andere normaal-eindigende transacties (of observators van stabiele toestanden van de database) dat kunnen waarnemen.

Dus wijzigingen aangebracht door een transactie T worden ongedaan gemaakt (indien T niet voltooid kan worden) zó dat andere transacties die wel tot voltooing komen, die wijzigingen niet kunnen waarnemen. Dit geldt zelfs na een abort van T of een crash van het systeem.

Geef een voorbeeld waarin de eigenschap *Atomicity* geschonden wordt.

Onleesbare tekst wordt fout gerekend.

Een transactie *Overboeking*, bestaande uit een actie *afschrijving* van de ene rekening en een actie *bijschrijving* op een andere rekening. Wanneer *Overboeking* zó zou worden uitgevoerd dat de ene actie wel wordt gedaan en de andere niet, dan is *Atomicity* geschonden.

Ander voorbeeld: Als $T_1 = w_1(x)$; ... en $T_2 = r_2(x)$ en de uitvoering is: $w_1(x)$; $r_2(x)$; $commit_2$; $abort_1$, dan is *Atomicity* geschonden.

Benoem de techniek die gebruikt wordt om *Atomicity* te waarborgen, en leg kort uit *hoe* die techniek werkt.

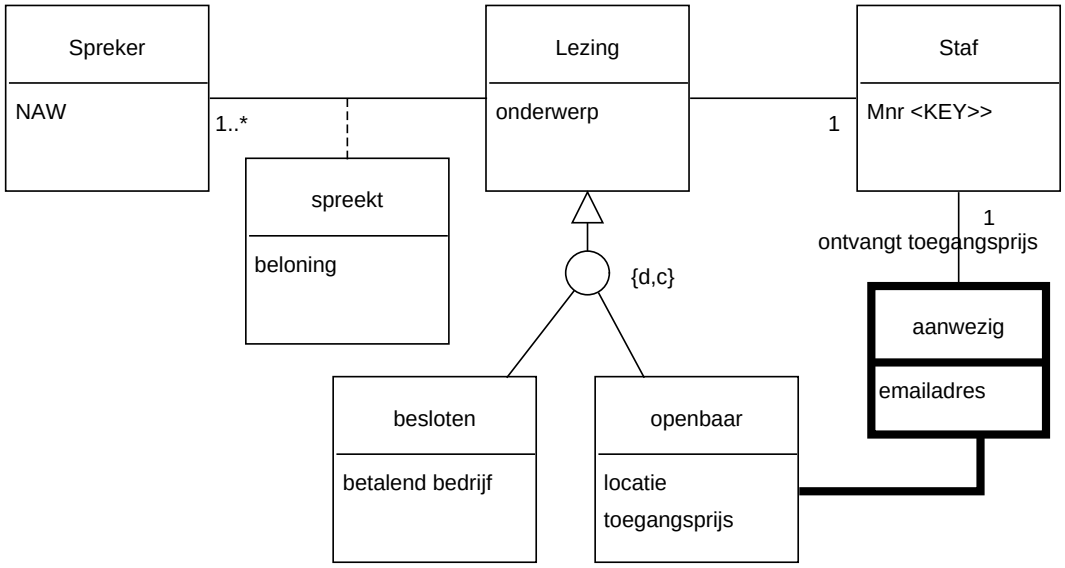
Onleesbare tekst wordt fout gerekend.

Logging (of preciezer: *write-ahead logging*).

Wanneer een transactie T voortijdig geeindigd is (door een abort van T of door een crash van het systeem), dan worden (onmiddellijk, of na opstarten na een crash) alle alreeds uitgevoerde acties van T teruggedraaid (*roll-back*); dat kan met behulp van de *before images* die bij iedere database-wijziging in de log geschreven worden vóórdat de actie op de database wordt uitgevoerd.

Toelichtingen

Toelichting bij antwoord 1.
Wellicht is de verzameling potentiële toehoorders zelf niet interessant; dit wordt bevestigd doordat er alleen over e-mailadressen van daadwerkelijke toehoorders wordt gesproken. Dit kan gemodelleerd worden door entiteittype *Toehoorder* te verbergen (transparant te tekenen of met een stippellijn of zoiets), zodat in feite alleen de daadwerkelijk aanwezige toehoorders een plaats hebben in het ERD. Een andere manier waarmee dit gemodelleerd kan worden is als volgt. Maak *aanwezig* tot een zwakke entiteit, met e-mailadres als partiële key en *openbaar* als identificerende entiteit:



Toelichting bij antwoord 2.
We hebben $c2$ óók opgenomen in D $(= S)$; die had eventueel achterwege gelaten kunnen worden. De eigenschap *not null* bij *precies* iedere foreign key is nodig om te voorkomen dat er volgens het database schema andere informatie opgeslagen kan worden dan volgens het ER-diagram. (Ieder attribuut in een primary key is per definitie al *not null*.)
De C instanties die een rol spelen (dus niet "alle mogelijke" C instanties) kunnen als volgt uit D verkregen worden:

select distinct $c1$, $c2$ from D;

Merk op dat S gelijk is aan relatie D (desgewenst met minder attributen). Als relatie zonder attributen $d1, d2$ (dus bestaande uit uitsluitend de keys van de deelnemende relaties B en C) kan S als volgt uit D verkregen worden:

select $a1$, $c1$ from D;

De 1 multipliciteit bij T is gemodelleerd door T in D te representeren. Relatie T (bestaande uit de keys van de deelnemende relaties A en D) is als volgt te verkrijgen uit de database:

select $a1A$, $(a1B, c1)$ from D;

Relatie R is op een standaard manier gemodelleerd door hem in B op te nemen; R (bestaande uit de keys van de deelnemende relaties) is als volgt uit de database te verkrijgen:

select $a.a1$, $b.a1$ from A a, B b where $a.a1 = b.a1$;

Toelichting bij antwoord 3.
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 4.
Een geheel andere decompositie van $\mathbf{R}$ wordt verkregen door de schending $D{\rightarrow}FG$ als uitgangspunt te kiezen. We krijgen dan $\mathbf{R}_1 = (DFG, \{D{\rightarrow}FG\})$ en $\mathbf{R}_2 = (ABCDE, \{AB{\rightarrow}CD,\ E{\rightarrow}D\})$, en in dit geval blijven alle FDs behouden! Van deze schema's staat $\mathbf{R}_1$ in BCNF en $\mathbf{R}_2$ niet.

Verdere decompositie van $\mathbf{R}_2$ aan de hand van $AB{\rightarrow}CD$ levert $\mathbf{R}_{2a} = (ABCD, \{AB{\rightarrow}CD\})$ en $\mathbf{R}_{2b} = (ABE, \{\,\})$; beide staan in BCNF, en $E{\rightarrow}D$ is niet behouden.

Verdere decompositie van $\mathbf{R}_2$ aan de hand van $E{\rightarrow}D$ levert $\mathbf{R}_{2a'} = (ED, \{E{\rightarrow}D\})$ en $\mathbf{R}_{2b'} = (ABCE, \{AB{\rightarrow}C\})$; beide staan in BCNF, en $AB{\rightarrow}D$ is niet behouden.

Een geheel andere decompositie van $\mathbf{R}$ wordt verkregen door de schending $E{\rightarrow}D$ als uitgangspunt te kiezen. We krijgen dan $\mathbf{R}_1 = (ED, \{E{\rightarrow}D\})$ en $\mathbf{R}_2 = (ABCEFG, \{AB{\rightarrow}CFG,\ E{\rightarrow}FG\})$. FDs $AB{\rightarrow}D$ en $D{\rightarrow}FG$ zijn niet behouden, en van deze schema's staat $\mathbf{R}_1$ in BCNF en $\mathbf{R}_2$ niet. Verdere decompositie van $\mathbf{R}_2$ aan de hand van $AB{\rightarrow}CFG$ levert $\mathbf{R}_{2a} = (ABCFG, \{AB{\rightarrow}CFG\})$ en $\mathbf{R}_{2b} = (ABE, \{\,\})$; beide staan in BCNF, en $E{\rightarrow}FG$ is niet behouden.

Verdere decompositie van $\mathbf{R}_2$ aan de hand van $E{\rightarrow}FG$ levert $\mathbf{R}_{2a'} = (EFG, \{E{\rightarrow}FG\}$ en $\mathbf{R}_{2b'} = (ABCE, \{AB{\rightarrow}C\})$; beide staan in BCNF, en $AB{\rightarrow}FG$ is niet behouden.

Toelichting bij antwoord 7.
Vervanging van het deel 'count $(s.n)$' door 'count (distinct $s.n$)' of 'count $(*)$' wijzigt de uitkomst niet, omdat in iedere groep geldt:
(i) er is hooguit één S-rij per waarde van $s.n$
want n is een key in S;
(ii) er is hooguit één O-rij per waarde van $s.n$
want per waarde van $s.n$:
er is precies een waarde van $o.b$ (vanwege 'group by $o.b$'), en
er is precies een waarde van $o.s$ (vanwege 'where $s.n = o.s$'), en dus
er is precies één O-rij (vanwege '(s, b) is key in O');
(iii) er is precies één C-rij per waarde van $s.n$
want per waarde van $s.n$:
er is precies één S-rij (zie eerder) en dus
er is precies éń waarde van $s.c$ en dus
er is precies één waarde van $c.n$ (vanwege 'where $c.n = s.c$'), en dus
er is precies één C-rij (vanwege 'n is een key in C').
Conclusie (i) is het bewijs dat 'count($s.n$)' en 'count (distinct $s.n$)' gelijkwaardig zijn.
Conclusies (i,ii,iii) zijn het bewijs dat 'count($s.n$)' en 'count $(*)$' gelijkwaardig zijn.

De having clause kan vervangen worden door de volgende extra conditie in de where clause:

exists (select $*$ from $C\ c1,\ S\ s1,\ O\ o1$

where $s1 \neq s$ and
$c.c = c1.c$ and $c1.n = s1.c$ and $s1.n = o1.s$ and $o1.b = o.b$)

(en met shunting kan de 'exists' weggewerkt worden en komen de $c1, s1, o1$ in de buitenste query). Voor deze oplossing wordt $2\frac{1}{2}$ punt afgetrokken, omdat de group by constructie niet benut wordt.