TECHNISCHE UNIVERSITEIT DELFT
Faculteit Elektrotechniek, Wiskunde en Informatica

Tentamen

IN2105/IN2410 Databases

Dinsdag 30 oktober 2007, 14:00 – 17:00

Dit tentamen bestaat uit 5 open vragen
Totaal aantal pagina's (inklusief deze pagina): 5

Bij dit tentamen hoort een appendix met een aantal nuttige plaatjes uit het boek en een aantal figuren waarnaar verwezen wordt in de opdrachten.

Opgave	Punten
1	20
2	15
3	30
4	20
5	15
Totaal	100

Alle deelopgaven binnen een opgave tellen even zwaar
Aantal punten nodig voor een voldoende: 55

Het gebruik van boeken, diktaten en van rekenmachines is niet toegestaan

Vul je naam, studienummer en studierichting in op ieder antwoordblad. Geef aan of het resultaat geadministreerd moet worden onder IN2105 of onder IN2410.

Dit tentamen omvat het materiaal uit de kollegesheets 2007-2008 (zie Blackboard), en de hoofdstukken 1 tot en met 8 (behalve sekties 3.6, 4.7 en 8.7), 10 en de sekties 11.1 en 11.2.2 uit hoofdstuk 11 uit het boek: Elmasri & Navathe, Fundamentals of Database Systems (Ed.5).

Opgave 1 (20 punten)

Geef voor elk van de hierna volgende vragen een SQL-query over de COMPANY database (het 'running example' uit het boek). Het schema van deze database wordt gegeven in Figuur 1 in de Appendix. Voor de syntax van SQL, zie Figuur 5 uit de Appendix.

a) Geef de namen (*Dname*) van alle afdelingen die een vestiging (*Dlocation*) in Bellaire hebben

b) Geef de som van de salarissen van de medewerkers die deelnemen aan een projekt in Houston

c) Geef de namen (*Dname*) van de afdelingen waar meer vrouwen dan mannen werken

d) Geef de *Ssn* van de medewerkers die
- minstens één familielid (*Dependent*) in de database hebben èn
- waarvoor geldt dat alle familieleden in de database vrouwelijk zijn

Opgave 2 (15 punten)

Geef voor elk van de hierna volgende vragen een query in de relationele algebra. Voor de operatoren van de relationele algebra zie Figuur 4 uit de Appendix. De onderliggende database is, als boven, de COMPANY database, waarvan het schema gegeven wordt in Figuur 1 in de Appendix.

a) Geef de namen (*Lname*) en de salarissen van alle afdelingschefs

b) Geef de namen (*Pname*) van de projekten die ofwel gesuperviseerd worden door de afdeling met *Dname* Research, ofwel waarin één of meer medewerkers participeren voor meer dan 10 uur.

c) Geef de namen (*Dname*) van de afdelingen die op alle projektlokaties (*Plocation* in PROJECT) een project superviseren.

Opgave 3 (30 punten)

Ontwerp een EER diagram voor de administratie van de tandartsenpraktijk SUM ("Spoelt U Maar").
In de eerste plaats wordt een administratie van de patienten bijgehouden. Per patient worden opgeslagen het (unieke) sofinummer, naam, adres, postcode en woonplaats, en (mogelijk meer dan één) telefoonnummer. Daarnaast is er voor iedere patient een uniek patientnummer.
Ook worden de behandelingen die patienten ondergaan geadministreerd. Iedere behandeling heeft een uniek identifikatienummer, en verder worden datum en tijdstip van de behandeling opgeslagen. Een behandeling bestaat uit een aantal verrichtingen. Een verrichting is bij voorbeeld een gaatje vullen in de hoektand links boven.
Per behandeling wordt voor iedere verrichting een uniek volgnummer bijgehouden. Verder wordt per verrichting bijgehouden om welke verrichting het gaat, over welk gebitselement het gaat (bij voorbeeld 'achterste verstandskies linksonder', hiervoor is een standaardcode), en het resultaat van de verrichting (bij voorbeeld 'mislukt', 'geslaagd', 'later nog controleren').
Iedere verrichting vindt plaats in een behandelkamer. De tandartspraktijk heeft een aantal behandelkamers, elk met een uniek nummer. De verrichtingen die onderdeel zijn van een behandeling hoeven zich niet allemaal af te spelen in dezelfde behandelkamer.
Daarnaast is er een administratie van de personeelsleden van de praktijk. Per personeelslid wordt bijgehouden het sofinummer, naam, adres, postcode en woonplaats, telefoonnummer(s) en het salaris. Alhoewel het niet wordt aangemoedigd, is het mogelijk dat personeelsleden zelf ook als patient bij de praktijk zijn aangesloten.
Van iedere verrichting wordt bijgehouden door welke personeelsleden die uitgevoerd is, bij voorbeeld door een van de tandartsen uit de praktijk in samenwerking met een assistente.
Niet iedere behandelkamer is geschikt voor elke verrichting. Ten behoeve van de planning wordt daarom ook bijgehouden in welke behandelkamer welk soort verrichting gedaan kan worden. Daartoe wordt van iedere mogelijke verrichting een unieke standaardcode bijgehouden, een omschrijving en de prijs per verrichting.

Maak een EER diagram voor de hierboven omschreven toepassing. Specificeer elke aanname die je verwerkt in je diagram en die niet uit de beschrijving hierboven afgeleid kan worden. Denk verder aan de eerste E in de afkorting EER, specialisatie en dat soort dingen. Tenslotte, je krijgt strafpunten voor elk sleutelattribuut dat je gebruikt dat niet genoemd wordt in de beschrijving hierboven.
Zie verder Figuur 3 uit de Appendix voor een overzicht van de ER-symbolen (en Figuur 2 voor enkele EER-symbolen).

Opgave 4 (20 punten)

Beschouw het EER diagram in Figuur 2 uit de Appendix. Vertaal dit diagram in een relationeel database schema. Noteer de tabellen zoals ik dat gedaan heb in Figuur 1 van de Appendix, en voeg (zoals in Figuur 1 gedaan is) voor elk foreign key – key paar een pijl toe van de foreign key naar de bijbehorende key. Denk erom de primaire sleutels te onderstrepen. Het is niet toegestaan primaire sleutels toe te voegen die niet als attribuut voorkomen in het EER diagram.

Ter verduidelijking, dit EER diagram kombineert een categorisatie met een specialisatie. Als uitgangspunt is het plaatje uit het boek, en de sheets, genomen waarin categorisatie behandeld werd.
Om je een idee te geven waar het allemaal over gaat geef ik een korte beschrijving. Auto's worden onderscheiden in personenauto's en vrachtauto's. Deze onderverdeling is niet disjunkt vanwege het feit dat er ook nog zoiets is als een bestelauto, die bij beiden thuishoort. Participatie van de entiteit AUTO is niet totaal omdat er bij voorbeeld ook nog autobussen en dergelijke bestaan.
Zowel personenauto's als vrachtauto's kunnen geregistreerd zijn en een kenteken hebben. Registratie is op naam van een EIGENAAR, die in dit geval alleen maar een persoon kan zijn (geen bank of bedrijf zoals in het boek en de sheets).
Elk adres van een eigenaar wordt uniek bepaald door de kombinatie postcode en huisnummer. Verschillende eigenaren kunnen hetzelfde adres hebben, de daarbij behorende telefoonnummers zijn soms verschillend, soms hetzelfde.
Auto's met kenteken zijn geregistreerd op naam van één eigenaar. Een eigenaar kan natuurlijk wel meer dan één geregistreerde auto hebben. Merk verder op dat de relatie "onderhoudt" tussen GARAGE en AUTO many to many is. Het attribuut 'KvKnummer' is het Kamer van Koophandelnummer dat voor ieder bedrijf uniek is.

Opgave 5 (15 punten)

Zij gegeven een relationeel schema R met attributen A, B, C, D, E en F.
Tussen deze attributen gelden de volgende funktionele afhankelijkheden:

$$
\begin{aligned}
A &\rightarrow B \\
CD &\rightarrow E \\
E &\rightarrow D \\
B &\rightarrow F
\end{aligned}
$$

a) Toon aan dat ACD een kandidaat sleutel is
b) Geef aan of er nog meer kandidaat sleutels zijn, en zo ja, welke.
c) Geef een verliesloze ("lossless") dekompositie van het bovenstaande schema naar een kollektie subschema's die allemaal in Boyce Codd normaalvorm staan.
Een voorbeeld van zo'n dekompositie is

(A,C,F) (B,C) (C,D,E)

(in dit geval jammer genoeg niet lossless en ook niet in Boyce Codd normaalvorm).
Nota Bene, het zal je niet lukken om alle funktionele afhankelijkheden te behouden. De funktionele afhankelijkheden tussen attributen die niet meer in één tabel voorkomen kun je buiten beschouwing laten.

Geef een onderbouwing van je antwoorden.

Einde van het tentamen

TECHNISCHE UNIVERSITEIT DELFT
Faculteit Elektrotechniek, Wiskunde en Informatica

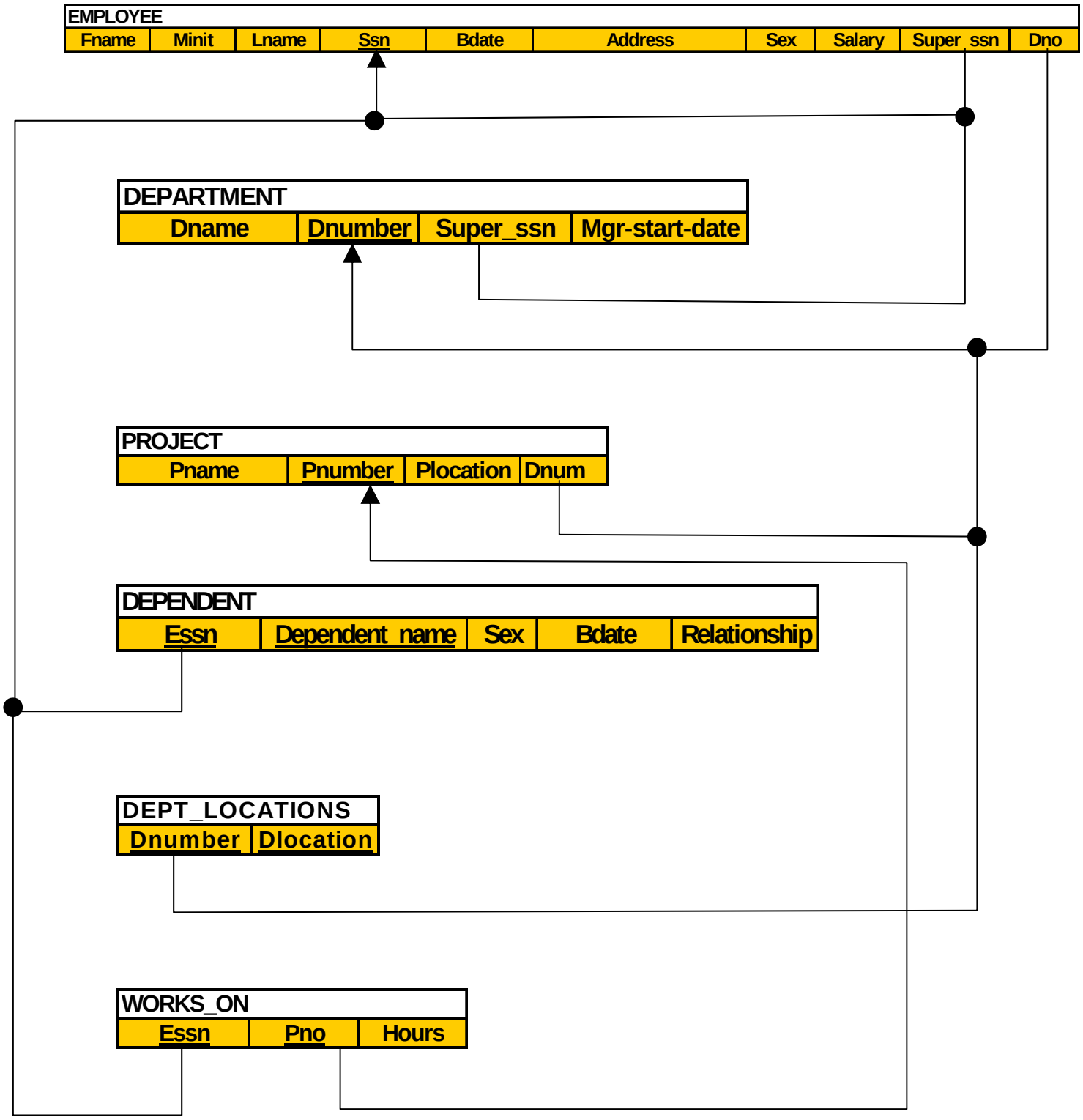
APPENDIX

behorend bij het tentamen

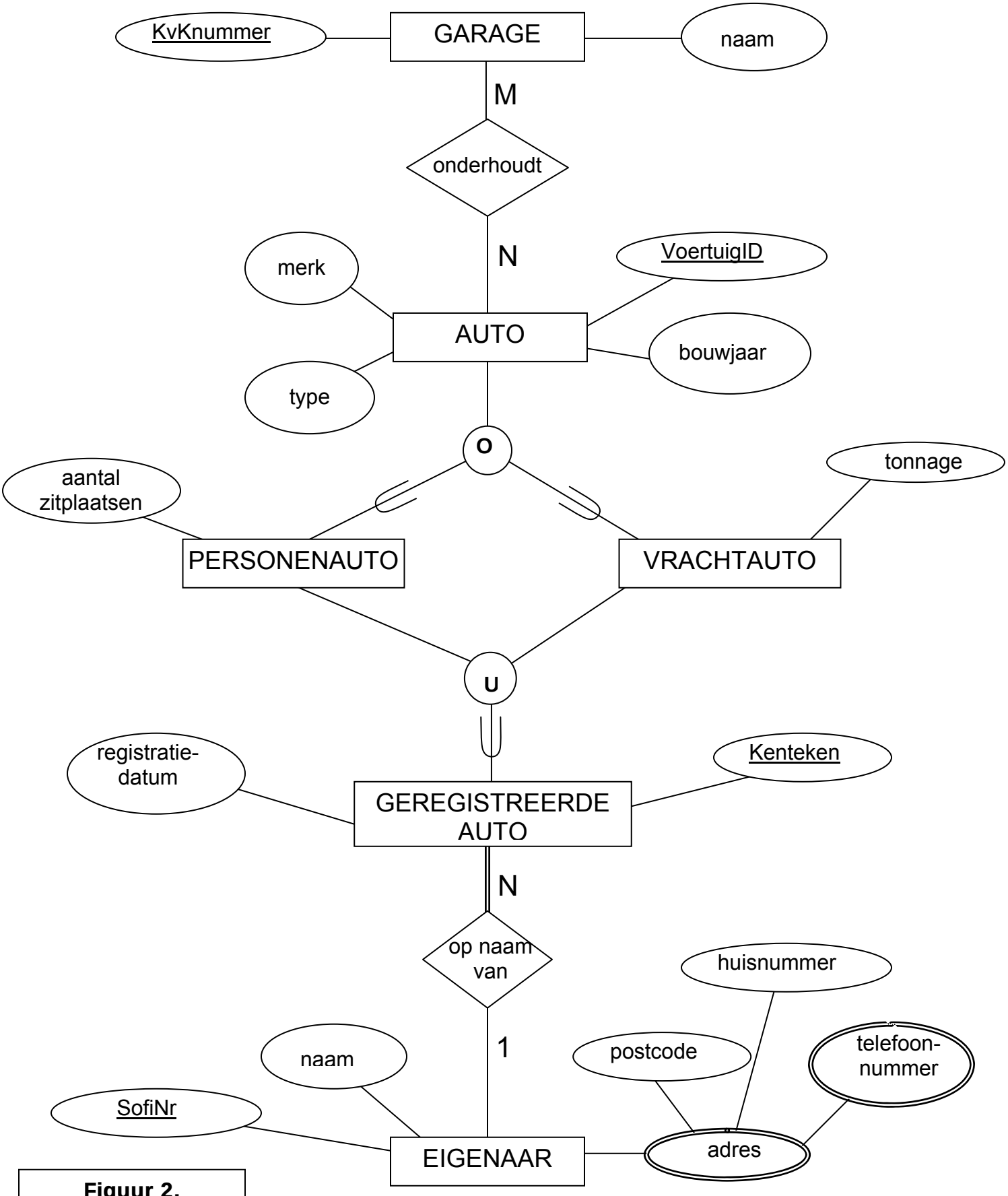
IN2105/IN2410 Databases

Dinsdag 30 oktober 2007, 14:00 – 17:00

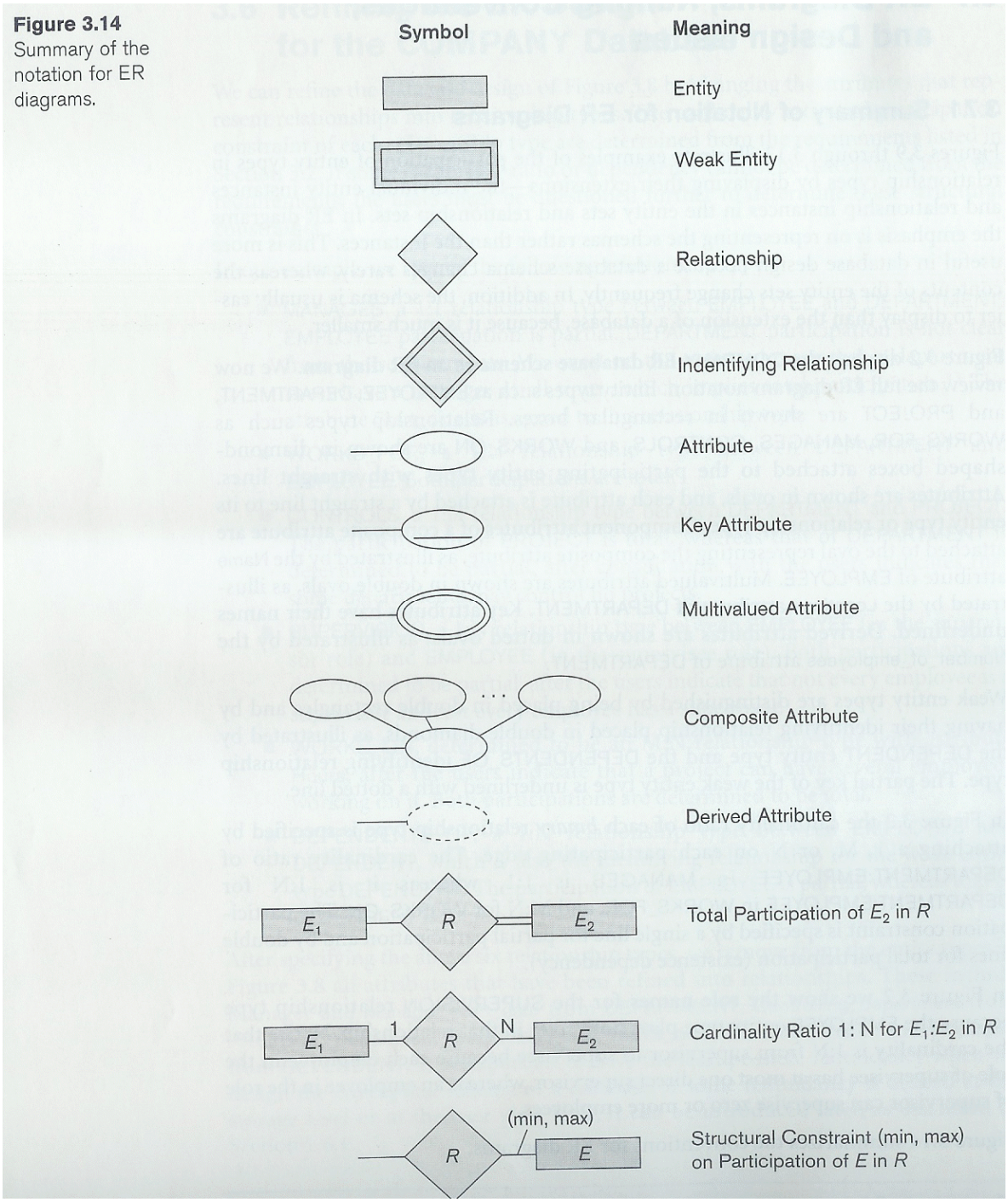
Totaal aantal pagina's (inklusief deze pagina): 6



Figuur 1. Schema van de COMPANY database. Zie Opgaven 1 en 2.



Figuur 2.
Voertuigregistratie
(zie Opgave 4)



Figuur 3. De ER symbolen en hun betekenis

Table 6.1
Operations of Relational Algebra

Operation	Purpose	Notation
SELECT	Selects all tuples that satisfy the selection condition from a relation R.	$\sigma_{<\text{selection condition}>}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{<\text{attribute list}>}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{<\text{join condition}>} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{<\text{join condition}>} R_2$, OR $R_1 \bowtie_{(<\text{join attributes 1}>),\ (<\text{join attributes 2}>)} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{<\text{join condition}>} R_2$, OR $R_1 *_{(<\text{join attributes 1}>),\ (<\text{join attributes 2}>)} R_2$ OR $R_1 * R_2$
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 that are not in R_2; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2.	$R_1 \times R_2$
DIVISION	Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

Figuur 4. De operatoren uit de relationele algebra

Table 8.2
Summary of SQL Syntax

CREATE TABLE <table name> (<column name> <column type> [<attribute constraint>] { , <column name> <column type> [<attribute constraint>] } [<table constraint> { , <table constraint> }])
DROP TABLE <table name>
ALTER TABLE <table name> ADD <column name> <column type>
SELECT [DISTINCT] <attribute list> FROM (<table name> { <alias> } | <joined table>) { , (<table name> { <alias> } | <joined table>) } [WHERE <condition>] [GROUP BY <grouping attributes> [HAVING <group selection condition>]] [ORDER BY <column name> [<order>] { , <column name> [<order>] }]
<attribute list> ::= (* | (<column name> | <function> (([DISTINCT] <column name> | *))) { , (<column name> | <function> (([DISTINCT] <column name> | *)) }))
<grouping attributes> ::= <column name> { , <column name> }
<order> ::= (ASC | DESC)
INSERT INTO <table name> [(<column name> { , <column name> })] (VALUES (<constant value> , { <constant value> }) { , (<constant value> { , <constant value> }) } | <select statement>)
DELETE FROM <table name> [WHERE <selection condition>]
UPDATE <table name> SET <column name> = <value expression> { , <column name> = <value expression> } [WHERE <selection condition>]
CREATE [UNIQUE] INDEX <index name> ON <table name> (<column name> [<order>] { , <column name> [<order>] }) [CLUSTER]
DROP INDEX <index name>
CREATE VIEW <view name> [(<column name> { , <column name> })] AS <select statement>
DROP VIEW <view name>

NOTE: The commands for creating and dropping indexes are not part of standard SQL2.

Figuur 5. SQL syntax

TECHNISCHE UNIVERSITEIT DELFT
Faculteit Elektrotechniek, Wiskunde en Informatica

UITWERKINGEN

behorend bij het tentamen

IN2105/IN2410 Databases

Dinsdag 30 oktober 2007, 14:00 – 17:00

Totaal aantal pagina's (inklusief deze pagina): 9

Opgave 1

a) Geef de namen (*Dname*) van alle afdelingen die een vestiging (*Dlocation*) in Bellaire hebben

```
SELECT Dname
FROM DEPARTMENT JOIN DEPT_LOCATIONS
                    ON DEPARTMENT.Dnumber = DEPT_LOCATIONS.Dnumber
WHERE Dlocation = 'Bellaire';
```

b) Geef de som van de salarissen van de medewerkers die deelnemen aan een projekt in Houston

```
SELECT SUM(Salary)
FROM EMPLOYEE
WHERE Ssn IN
  (SELECT Essn
    FROM WORKS _ON, PROJECT
    WHERE Pno = Pnumber AND Plocation = 'Houston');
```

of:

```
CREATE VIEW Houston AS
  (SELECT DISTINCT Essn
    FROM WORKS_ON JOIN PROJECT ON Pno = Pnumber
    WHERE Plocation = 'Houston');

SELECT SUM(Salary)
FROM EMPLOYEE JOIN Houston ON Essn = Ssn;

DROP VIEW Houston;
```

c) Geef de namen (*Dname*) van de afdelingen waar meer vrouwen dan mannen werken

```
CREATE VIEW Vrouwen(Dno, Aantal) AS
  (SELECT Dno, COUNT(Ssn)
   FROM EMPLOYEE
   WHERE Sex = 'F'
   GROUP BY Dno);

CREATE VIEW Mannen(Dno, Aantal) AS
  (SELECT Dno, COUNT(Ssn)
   FROM EMPLOYEE
   WHERE Sex = 'M'
   GROUP BY Dno);

SELECT Dname FROM DEPARTMENT, Vrouwen, Mannen
WHERE DEPARTMENT.Dnumber = Vrouwen.Dno
  AND DEPARTMENT.Dnumber = Mannen.Dno
  AND Vrouwen.Aantal > Mannen.Aantal;

DROP VIEW Vrouwen;

DROP VIEW Mannen;
```

d) Geef de *Ssn* van de medewerkers die

- minstens één familielid (*Dependent*) in de database hebben èn
- waarvoor geldt dat alle familieleden in de database vrouwelijk zijn

```
SELECT Essn FROM DEPENDENT
WHERE Essn NOT IN
  (SELECT Essn FROM DEPENDENT
    WHERE Sex <> 'F');
```

Opgave 2

a) Geef de namen (*Lname*) en de salarissen van alle afdelingschefs

DepChefs(Ssn) ← π_{Super_ssn} (DEPARTMENT)
Result ← $\pi_{Lname,Salary}$ (EMPLOYEE * DepChefs)

b) Geef de namen (*Pname*) van de projekten die ofwel gesuperviseerd worden door de afdeling met *Dname* Research, ofwel waarin één of meer medewerkers participeren voor meer dan 10 uur.

ResearchDept(Dnum) ← $\pi_{Dnumber}(\sigma_{Dname='Research'}$(DEPARTMENT))
ResearchProjects ← π_{Pname}(PROJECT * ResearchDept)
PnoMoreThan10(Pnumber) ← $\pi_{Pno}(\sigma_{Hours>10}$(WORKS_ON))
PnameMoreThan10 ← π_{Pname}(PROJECT * PnoMoreThan10)
Result ← ResearchProjects U PnameMoreThan10

c) Geef de namen (*Dname*) van de afdelingen die op alle projektlokaties (*Plocation* in PROJECT) een project superviseren.

DeptLoc(Dnumber,Plocation) ← $\pi_{Dnum,Plocation}$(PROJECT)
AllLocations ← $\pi_{Plocation}$(PROJECT)
DeptAllLocations ← DeptLoc ÷ AllLocations
Result ← π_{Dname}(DEPARTMENT * DeptAllLocations)

Opgave 3

Zie Figuur 1.

Ik heb de volgende aannames:

- Een behandeling betreft één patient
- Er zijn patienten die geen behandeling gehad hebben
- Er zijn personen die noch personeelslid noch patient zijn
- Een verrichting speelt zich af in één behandelkamer
- Niet ieder type verrichting is ook uitgevoerd
- Er zijn verrichtingen die in geen enkele behandelkamer kunnen worden uitgevoerd
- In een behandelkamer kan op zijn minst één verrichting uitgevoerd worden en in het algemeen meer dan één
- In elke behandelkamer is minstens één verrichting uitgevoerd
- Bij iedere verrichting is minstens één personeelslid betrokken
- Er zijn personeelsleden die geen verrichtingen doen

Opgave 4

Zie Figuur 2.

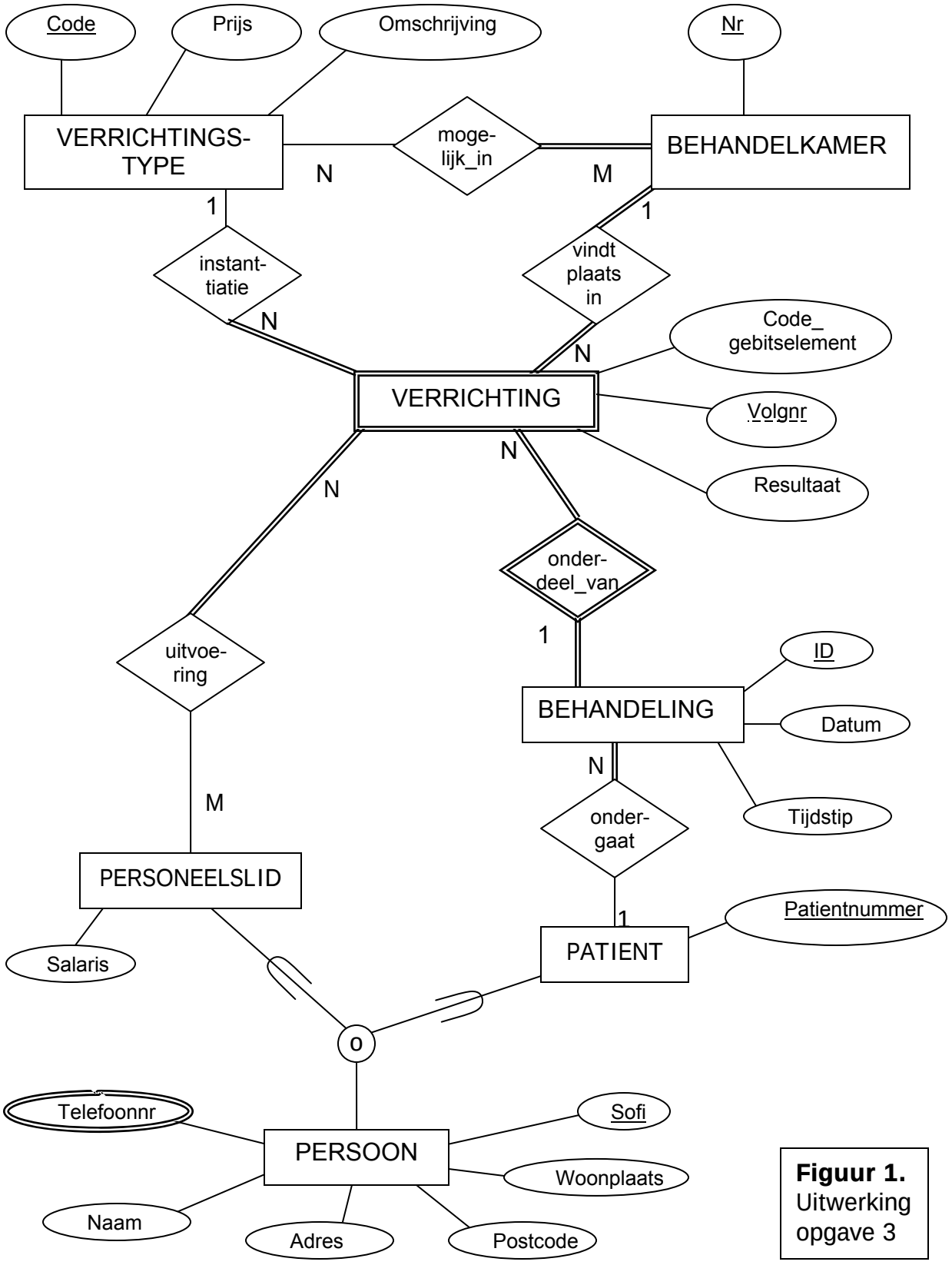
De uitwerking van de entiteiten GARAGE en AUTO, met de relatie ONDERHOUDT, is standaard. Dat geldt ook voor de specialisaties PERSONENAUTO en VRACHTAUTO van AUTO.
De categorie GERIGISTREERDE AUTO heb ik opgelost door gebruik te maken van de key 'kenteken' die hier dienst doet als pseudokey. Daarom moet het attribuut 'kenteken' in de tabel GEREGISTREERDE AUTO primary key zijn. Dat betekent dat het attribuut 'VoertuigID' in deze tabel geen primary key is, alleen foreign key. Je kunt in plaats daarvan hetzelfde spel spelen met het attribuut 'VoertuigID' in PERSONENAUTO, VRACHTAUTO en GEREGISTREERDE AUTO, maar dan moet je wel in PERSONENAUTO en VRACHTAUTO het attribuut 'VoertuigID' twee keer opnemen, een keer als primary key en foreign key naar AUTO, en een maal als foreign key naar GEREGISTREERDE AUTO, die NULL kan zijn. Dat vond ik niet zo elegant. Tenslotte kun je eventueel de foreign key 'VoertuigID' weglaten in GEREGISTREERDE AUTO, maar dan wordt het wel moeilijker om de geërfde attributen 'Merk', 'Type' en 'Bouwjaar', gedefinieerd in de superklasse AUTO terug te vinden.
Het geneste meervoudige attribuut 'telefoonnummer' in de entiteit EIGENAAR heb ik 'uitgeprogrammeerd' door allereerst van het meervoudige attribuut 'Adres' een eigen entiteit te maken, dat zelf dan weer een meervoudig attribuut 'telefoonnummer' heeft, en dat laatste attribuut vervolgens zelf ook weer tot een zelfstandige entiteit te promoveren. De bijbehorende 1:N relaties heb ik op de standaardmanier uitgewerkt.
Dit levert een behoorlijke overhead op. Het is het overwegen waard om die te elimineren door in de tabel ADRES een key 'AdresID' op te nemen en in de tabel TELEFOON een foreign key op te nemen die daarnaar verwijst. Maar in de opgave stond dat dat niet was toegestaan...

Opgave 5.

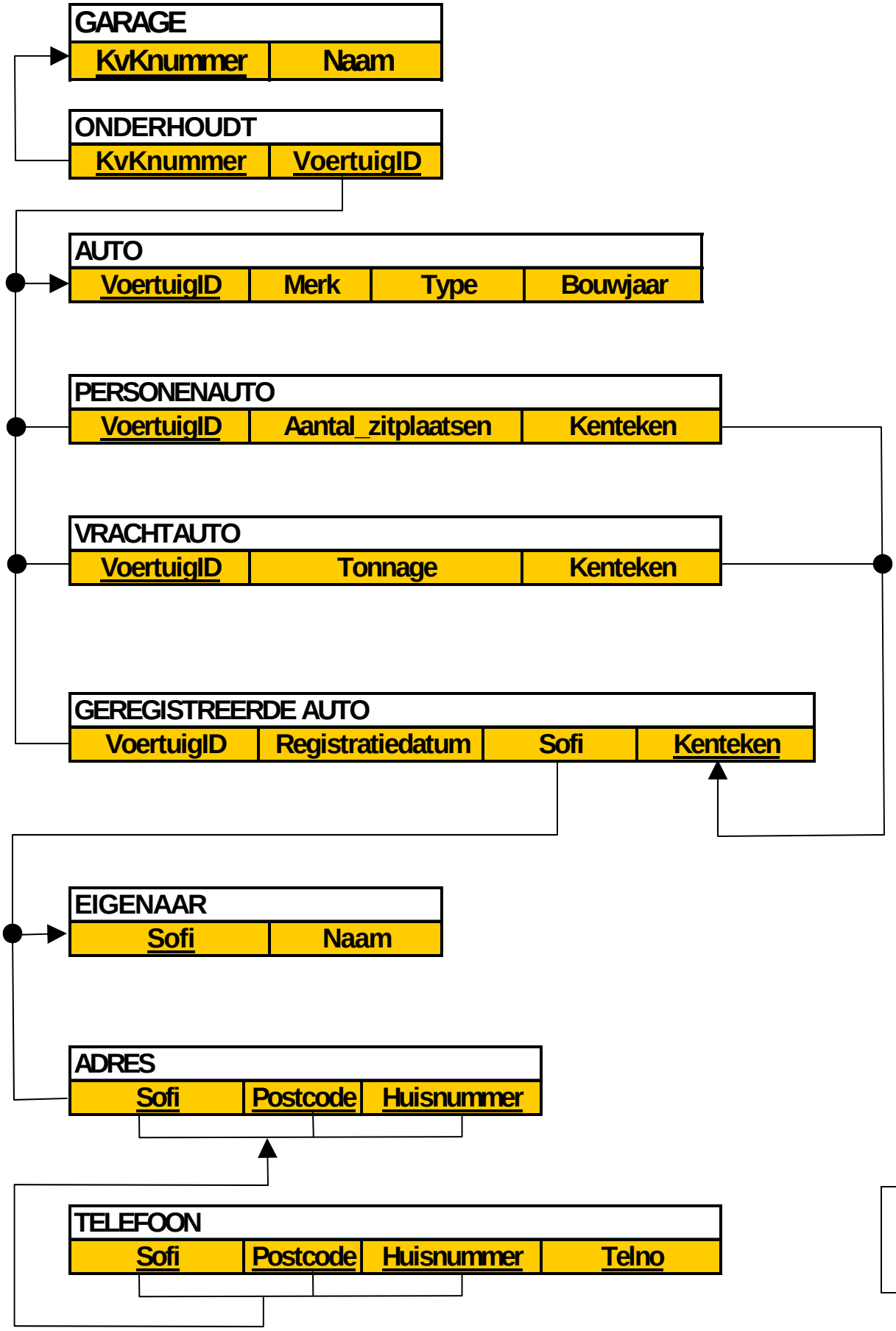
a) Eerst (ACD)+ bepalen om te kijken of dit een superkey is. Via A → B krijg ik (ABCD), en via CD → E krijg ik (ABCDE), en tenslotte via B → F krijg ik (ABCDEF), dus ACD is een superkey.
Nu nog aantonen dat deelverzamelingen van ACD geen superkeys zijn.
Allereerst (AC). Dit is geen superkey omdat je uit (AC) niet D en E kunt afleiden.
Verder is (AD) ook geen superkey omdat je C en E niet kunt bereiken. Tenslotte is (CD) ook geen superkey omdat je A, B en F niet kunt krijgen.

b) Iedere superkey moet A en C bevatten, omdat deze attributen niet voorkomen in het rechterlid van een funktionele afhankelijkheid. Als je (AC)+ bepaalt dan zie je dat dat de attribuutverzameling ABCF is. Dus om een superkey te maken moet je ofwel D en E toevoegen, of misschien één van de twee.
Als je D toevoegt krijg je ACD en dat is een kandidaat sleutel zoals in a) vastgesteld is. Als je E toevoegt krijg je via de regel E → D ook D erbij. Dat betekent dat ACE ook een superkey is. Dus is ACE ook een kandidaatsleutel. Dit zijn alle mogelijkheden.

c) Mijn oplossing houdt zoveel mogelijk funktionele afhankelijkheden in stand. Alle 4 de funktionele afhankelijkheden zorgen ervoor dat de oorspronkelijke relatie niet in Boyce Codd normaalvorm staat. Ik ga de overtredingen een voor een wegwerken op de standaardmanier.
Eerst B → F. Dit levert de twee subschema's (A,B,C,D,E) en (B,F) op waarvan de laatste Boyce Codd is.
Dan A → B. Dit dekomponeert het eerste subschema naar (A,C,D,E) en (A,B) waarvan het laatste weer Boyce Codd is.
Vervolgens CD → E. Het eerste subschema wordt opgedeeld in (A,C,D) en (C,D,E) en nu staat het eerste subschema in BC normaalvorm.
Tenslotte E → D, dat ervoor zorgt dat (C,D,E) niet Boyce Codd is. De dekompositie is hier (C,E) en (D,E).
Dus de totale dekompositie is (A,B) (A,C,D) (B,F) (C,E) (D,E)



Figuur 1. Uitwerking opgave 3



Figuur 2. Uitwerking Opgave 4