Databases 1, Labsessie 1

Procedure

Tijdens de labsessie leg je je collegekaart op de bank/tafel zodat de instructeur of studentassistent je aanwezigheid kan noteren.

Tijdens de labsessie worden uitgereikte opgaven (zoals deze) gemaakt. Een instructeur en studentassistent zijn beschikbaar om vragen te beantwoorden, en zij zullen ook af en toe uitwerkingen tonen of iets aan het bord uitleggen.

Aan het eind van de labsessie krijg je huiswerkopgaven. Die dienen tegen de volgende colstructie te worden gemaakt en ingeleverd.

1 Voorkennis

Het vak Databases 1 gaat vooral over de hoofdstukken 3, 4 en 7 van het boek **Database System Concepts** van Silberschatz e.a. Noodzakelijke voorkennis zijn de vakken ISO (2R290) en Logica en verzamelingen (2IT10), afgekort L&V. Van dat laatste vak is het belangrijk dat je de regels kunt toepassen en logische formules kunt begrijpen en zelf opstellen. In de informatica, en dus ook bij het omgaan met databases is het belangrijk dat je precies vastlegt wat de bedoeling is; daarvoor is de taal van de logica een ideaal hulpmiddel. Bij het vak ISO heb je misschien al vast kunnen stellen dat SQL veel weg heeft van de logica.

Hier volgt een test voor je L&V-vaardigheden die voor DB1 nodig zijn.

Ingangstoets DB1

De variabelen x, y stellen personen voor. De atomaire predikaten zijn:

$M(x)$	x is een man
$K(x,y)$	x is een kind van y
$G(x,y)$	x en y zijn getrouwd

We eisen dat een persoon hetzij man hetzij vrouw is.

1 Druk de volgende uitspreken uit als logische formules.
   a Jan is een zoon van Ans,
   b Kees is de schoonvader van Jan,
   c Klazien is de enige schoondochter van Ans,
   d Erik en Wil zijn halfbroers.

2 Formuleer de volgende uitdrukkingen zo kernachtig mogelijk in natuurlijke taal.
   a $\neg M(b) \wedge (\exists x : K(a,x) : K(x,b))$,
   b $(\forall x, y : G(x,y) : M(x) \iff \neg M(y))$

3 Bewijs de equivalentie van de volgende twee uitdrukkingen en geef de betekenis:
   $(\forall x, y : (\exists z :: K(x,z) \wedge K(y,z)) : \neg G(x,y))$,
   $(\forall x, y : G(x,y) : (\forall z : K(x,z) : \neg K(y,z)))$.

2 Structuur van een database

In databases worden grote aantallen gegevens op een efficiënte manier opgeslagen. De kleinste eenheid van informatie is een *record*. Een record is een functie[1] met als domein een eindige verzameling *labels* of attribuutnamen. Bij elk label hoort een *waarde*. We kunnen een record t definiëren als volgt: $t = \{naam \mapsto Amalia, geboortedatum \mapsto 7/12/2003, geslacht \mapsto V\}$. De verschillende componenten van t kunnen weer verkregen worden door *applicatie* van een label uit het domein; bijvoorbeeld $t[naam] = Amalia$. Bij applicatie verdwijnt het label en blijft de waarde over. Een andere operatie is *projectie*; projectie van een record op een deelverzameling van zijn labels geeft een (kleiner) record. Voorbeeld: $t[\{naam, geslacht\}] = \{naam \mapsto Amalia, geslacht \mapsto V\}$ en $t[\{naam\}] = \{naam \mapsto Amalia\}$. Merk het verschil op tussen $t[naam]$ (een waarde) en $t[\{naam\}]$ (een record).

Een *tabel* is een verzameling[2] van records met het hetzelfde domein. Dit wordt in tabelvorm weergegeven met de labels bovenaan. Laat T de volgende tabel zijn.

naam	*geboortedatum*	*geslacht*
Amalia	7/12/2003	V
JanPaul	18/5/1920	M
Lotte	1/4/1987	V

In de relationele algebra worden er operatoren op tabellen gegeven, zoals bijvoorbeeld de (tabel)projectie. Zo is $\Pi_{naam}(T)$ gedefinieerd als $\{t[\{naam\}] \mid t \in T\}$.

Een database is verzameling *variabelen*, die op elk moment een tabel als waarde hebben. Een burgerlijke stand database kan bijvoorbeeld bestaan uit de variabelen *personen* met huidige waarde T en *huwelijken* met als waarde de lege tabel. Als een huwelijk wordt gesloten krijgt *huwelijken* een nieuwe waarde, bijvoorbeeld:

naamman	*naamvrouw*	*datum*
JanPaul	Amalia	22/05/2022

Met deze afspraken kunnen we alle nodige definities en begrippen over databases weergeven in de L&V notatie. Als de vertaling van "gewone" begrippen naar die notatie goed beheerst wordt zijn databases niet moeilijk meer. We sluiten af met een paar oefeningen om de smaak te pakken te krijgen. Vertaal naar of uit predikaten:

1 $(\forall t : t \in huwelijken : (\exists u : u \in personen : u.naam = t.naamman))$,
2 $(\forall t, u : t \in personen \wedge u \in personen \wedge t \neq u : t[naam] \neq u[naam])$,
3 Dezelfde man mag op dezelfde datum maar één keer trouwen
4 Er bestaan geen ongetrouwde vrouwen.

[1] In afwijking van het boek, waar een record een lijst is
[2] In implementaties soms een lijst of bag

3 Database Design

1. Geef alle mogelijke functionele afhankelijkheden die in de persoonstabel T (zie vorige pagina) **niet** gelden.
2. Beschouw de tabel van Figure 7.1. Geef aan welke functionele afhankelijkheden met link en rechts een singleton (1 attribuut) in deze instance **wel** gelden.

4 Relationele Algebra

Beschouw het database schema van Figure 3.9 (het bank schema).

1. Vertaal naar de relationele algebra:
   (a) "Geef de filiaal-namen van alle leningen."
   (b) "Geef de filiaal informatie van alle filialen met assets groter dan 1.000.000."
   (c) "Geef de namen van alle klanten die in Eindhoven wonen."
   (d) "Geef de filiaal-namen van alle rekeningen en van alle leningen (samen in één lijst)."
   (e) "Geef de namen van alle filialen waar geen lening is afgesloten."
2. Vertaal van relationele algebra naar het Nederlands:
   (a) $\Pi_{\text{account}-\text{number}}(\sigma_{\text{balance}>10.000}(account))$
   (b) $\Pi_{\text{customer}-\text{city}}(\text{customer}) - \rho_{\text{d(customer}-\text{city)}}(\Pi_{\text{branch}-\text{city}}(\text{branch}))$
   (c) $\Pi_{\text{customer}-\text{name}}(\sigma_{\text{account.account}-\text{number}=\text{depositor.account}-\text{number}}(\sigma_{\text{balance}>10.000}(\text{account}) \times \text{depositor}))$