Naam en voorletters: Ident.nr

TECHNISCHE UNIVERSITEIT EINDHOVEN

Faculteit Wiskunde en Informatica

Tentamen Databases 1, 2M400, 28 juni 2004.

Alle uitwerkingen van de opgaven **moeten** worden ingevuld in de daarvoor bestemde vrije ruimte bij de tentamenvragen. **Alleen wat op deze formulieren is ingevuld wordt nagekeken. Andere papieren worden meteen vernietigd. Ze worden in geen geval nagekeken**. De ruimte die is voorzien is in elk geval voldoende voor het correcte antwoord, in een normaal leesbare lettergrootte. Alles wat onvoldoende leesbaar is wordt fout gerekend! Elk antwoord moet staan in de ruimte bij de vraag, tenzij verwezen is naar de laatste (witte) pagina.

Dit is een "open boek" tentamen. Je mag gebruik maken van boeken en aantekeningen. Alle niet door jezelf geschreven materiaal dient te dateren van vóór de aanvang van dit tentamen. Het gebruik van communicatieapparatuur en computers is niet toegestaan.

Dit tentamen bestaat uit twee delen. Er is een "pre"-tentamendeel en een "hoofd"-tentamendeel. Wanneer je geen vrijstelling voor het pre-tentamendeel hebt, dien je dit deel het eerst te maken. Het bestaat uit drie vragen, die alle drie correct beantwoord dienen te worden, voordat het hoofdtentamendeel wordt nagekeken. Wees dus zorgvuldig in het beantwoorden van het eerste deel. Daarna kun je verder gaan met de overige vragen. Kandidaten met een vrijstelling kunnen meteen aan het hoofddeel beginnen. Wie vrijstelling heeft en wie niet staat op het forum vermeld. Mocht je twijfelen of je vrijstelling hebt, maak dan het pretentamen!

Hier is een recept voor het met succes afronden van dit tentamen:

1. Bestudeer eerst alle vragen, en stel een lijstje op met de volgorde waarin je de vragen wil beantwoorden.
2. Begin met het beantwoorden van de vragen waarvan je zelf denkt dat je ze het beste kunt. Werk zorgvuldig. Het is niet erg om wegens tijdgebrek aan het eind een vraag open te moeten laten. Kijk ook naar het aantal punten per vraag. Begin met vragen die met de minste moeite en het kleinste risico zoveel mogelijk punten opleveren.
3. Denk bij elke vraag eerst goed na over het "type" van de vraag: welk deel van het collegemateriaal is van belang bij het beantwoorden van de vraag? Bij een query: wat voor soort query is het? Bij afleidingsregels: welke "richting" moet ik bewijzen? etc.
4. Werk elke vraag eerst op een kladblaadje uit en schrijf het antwoord dan over op het invulblad. **Wacht hiermee niet tot het einde want losse bijgevoegde papieren worden niet nagekeken! Sterker: ze worden door de surveillant geweigerd, en als hij/zij ze toch zou aannemen worden ze door de corrector meteen vernietigd!** Je mag bij een vraag verwijzen naar de laatste pagina, maar als die vol is houdt het op!
5. Als je nog 15 a 20 minuten hebt, begin dan niet meer aan een nieuwe vraag (die je niet ligt en daarom hebt uitgesteld tot het einde) maar kijk de vragen na die je al hebt beantwoord. Bij queries: vertaal ze zonder naar de opgave te kijken eens terug, en kijk dan of wat uit die vertaling komt ook de opgave was. Bij de afhankelijkheden of decompositie, ga na of je antwoord wel het antwoord is op de vraag die is gesteld: heb je de juiste "richting" bewezen? Heb je geen veronderstellingen gemaakt die de algemeenheid van je uitwerking schaden, voldoet bij een decompositie je resultaat wel aan de gevraagde normaalvorm, etc.
6. Verifieer dat je op elke pagina je naam en identiteitsnummer hebt geschreven.

Naam en voorletters: Ident.nr

In dit tentamen wordt gebruik gemaakt van een **reisadministratie-database**. Ze dient om een deel van de administratie van dienstreizen voor een universiteit te kunnen voeren. De database heeft de volgende tabellen (met onderlijnde sleutelattributen):

Reis(reisnr, bestemming, motivering, duur)
Reisdeel(reisdeelnr, reisnr, vervoermiddel, vertrekdag, startplaats, aankomstdag, bestemming)
Reiziger(reisnr, naam, faculteit)
Aanvraag(reisnr, faculteit, kostenplaats, naam, datum, goedgekeurd, geannuleerd)
Ticket(reisnr, reisdeelnr, datum, vertrekpunt, vertrektijd, aankomstpunt, aankomsttijd, vervoerder, aantal, prijs)

Korte beschrijving:
Elke reis heeft een uniek reisnummer, een bestemming (bijvoorbeeld een plaats), motivering (bijvoorbeeld het "bijwonen van de AH2004 conferentie") en duur (in dagen).
Een reis bestaat uit één of meer delen, elk met hun eigen startplaats en bestemming (plaats), vertrek- en aankomstdata en vervoermiddel. Tijdens een reis kunnen dus ook andere plaatsen dan de reisbestemming bezocht worden. Reisdelen hebben een reisdeelnummer (1, 2, 3...)
Een reis kan door meer dan één persoon gemaakt worden en een persoon kan meerdere reizen maken. Een reiziger (lees "deelname van een persoon aan een reis") wordt gekenmerkt door een reisnr, een (familie)naam en een faculteit.
Voor een reis dient een aanvraag ingediend te worden. (Wanneer er meer reisdeelnemers zijn dienen zij samen één aanvraag in.) Deze aanvraag wordt op een bepaalde datum voorafgaand aan de reisperiode ingediend. De reisperiode wordt gegeven door de vertrekdag voor het eerste reisdeel en de aankomstdag van het laatste deel van die reis. De aanvraag komt ten laste van een kostenplaats van de faculteit, waarbij deze ingediend wordt, en dient door een bevoegde persoon (naam) van diezelfde faculteit goedgekeurd te worden. Het attribuut "goedgekeurd" heeft de waarde NULL tot de uitslag voor de goedkeuring bekend is. De aanvraag kan eventueel ook nog geannuleerd worden. Het attribuut geannuleerd heeft als standaardwaarde "nee".
Voor sommige reisdelen worden de tickets van tevoren aangeschaft. Van een ticket wordt de datum van geldigheid, de plaats van vertrek en die van aankomst (station, vliegveld) genoteerd en indien van toepassing de vertrek- en aankomsttijd (op de vertrek- en aankomstdag). Indien niet van toepassing zijn die tijden NULL. Bij elk ticket hoort een vervoerder (KLM, NS), een prijs (in euro's) en het aantal (tenminste één) dat ervan is aangeschaft.
Een paar beperkingen of eigenschappen die gelden in deze database (en waarvan je soms mag of zelfs moet gebruik maken) zijn:

- Bij elk kostenplaats(nummer) hoort één faculteit.
- Een tekenbevoegde persoon hoort bij één faculteit. Hij mag slechts goedkeuring geven aan reizen die ten laste komen van één faculteit, maar wel voor reizigers van eender welke faculteit. Zijn naam hoeft **niet uniek** te zijn. (Twee faculteiten kunnen elk een tekenbevoegde persoon hebben met dezelfde naam.) Een faculteit kan verschillende tekenbevoegde personen hebben, en ook voor elke kostenplaats kunnen verschillende tekenbevoegde personen zijn.
- Elke faculteit van de universiteit heeft reizigers die in de database voorkomen.
- Elke reis moet een reisdeel hebben met als bestemming de bestemming van de hele reis.
- Elk ticket moet overeenkomen met een reisdeel (van dezelfde reis) maar er mogen reisdelen zijn zonder ticket.

Lees de bovenstaande beschrijving aandachtig. Wanneer je de werking van de reisadministratie niet correct en volledig begrijpt zul je fouten maken in verscheidene tentamenvragen!

Naam en voorletters: Ident.nr

Je mag het eerste vel losscheuren van de rest. Alle andere pagina's (dus vanaf deze pagina) moeten aan elkaar geniet blijven.

Algemene informatie (heeft geen enkele invloed op het cijfer):

Mijn colstructeur was: (aankruisen of omcirkelen svp)

De Bra Aerts Voorhoeve n.v.t.

Mijn lab-instructeur was: (aankruisen of omcirkelen svp)

Frasincar Vdovjak Thiran n.v.t.

n.v.t. betekent: ik volgde weinig of geen colstructies of labsessies

Pre-tentamendeel

Belangrijke hint (niet alleen voor dit deel trouwens): **Alle vragen kunnen redelijk eenvoudig en bondig beantwoord worden. Als je denkt dat het antwoord heel ingewikkeld en lang wordt dan is er veel kans dat je verkeerd bezig bent.**

1. Beschrijf de volgende vraag over de reisadministratie-database in de relationele algebra:
   "Geef de bestemmingen van alle reisdelen met 11-09-2001 als vertrekdatum."

   Antwoord:

   $$\Pi_{bestemming}(\sigma_{vertrekdag='11-09-2001'}(reisdeel))$$

2. Beschrijf de volgende vraag over de reisadministratie-database in SQL:
   "Geef de reisnummers van reizen waar een ticket bij hoort met een datum die vóór de vertrekdag is."
   Je mag aannemen dat je data met < kunt vergelijken.
   Zorg ervoor dat elk reisnummer niet meer dan één keer in die lijst voorkomt.

   ```
   Select distinct D.reisnr
   From Reisdeel as D, Ticket as T
   Where T.reisnr = D.reisnr
   and T.datum < D.vertrekdag
   and D.reisdeelnr = 1
   ```

   Je mag ook `D.reisdeelnr = T.reisdeelnr` schrijven in plaats van `D.reisdeelnr = 1`. Dit geeft een iets andere betekenis: ticket datum vóór de vertrekdag van de deelreis (in plaats van de hele reis). Niet acceptabel is alleen gelijkheid van reisnr eisen: data van tickets van de eerste reisdag komen altijd vóór de vertrekdag van de latere reisdagen.

Naam en voorletters: Ident.nr

3. Beschouw een relationeel schema R met attributen {A, B, C}, elk met een groot domein zoals integers of strings, en de verzameling functionele afhankelijkheden F = {A $\rightarrow$ B, B $\rightarrow$ C, C $\rightarrow$ A}. Is dit schema in BCNF? Leg kort uit waarom dit wel of niet zo is. (Zonder uitleg wordt deze vraag fout gerekend.)

   Het schema is in BCNF. Alle afhankelijkheden in F zijn namelijk sleutelafhankelijkheden (d.w.z. hebben een superkey aan de linkerkant).
   $A^+ = B^+ = C^+ = ABC$.

Einde pre-tentamendeel.

Hoofd-tentamendeel

1. Druk de volgende vraag over de reisadministratie-database uit in de relationele algebra, in de tupel calculus en in SQL: (in **alle drie de formalismen** dus)

   "Geef de faculteiten bij wie er (nog) geen enkele reis is aangevraagd."

   Zorg er in de SQL versie voor dat een faculteit niet meer dan één keer in die lijst voorkomt. (In de algebra en de tupel calculus is dit automatisch altijd het geval.)
   Als je de reisadministratie-beschrijving niet zorgvuldig gelezen hebt kun je denken dat het antwoord op deze vraag altijd leeg moet zijn. Als je dit denkt moet je de beschrijving opnieuw en nu grondiger bestuderen.

Antwoord:
De verzameling faculteiten vinden we door te kijken naar de faculteiten waarvan personen deelnemen aan een reis. De faculteiten waarbij een reis is aangevraagd halen we uit de aanvragen. Het antwoord wordt gegeven door het verschil:

RA: $\Pi_{faculteit}(Reiziger) - \Pi_{faculteit}(Aanvraag)$

TC:
{ t | $\exists$r $\in$ reiziger (r[faculteit] = t[faculteit] $\wedge$ $\neg\exists$ a $\in$ aanvraag (a[faculteit]=r[faculteit]))}

SQL:
```
Select faculteit From reiziger
Except
Select faculteit From aanvraag
```

Fout:
Een overbodig verband leggen tussen reiziger en aanvraag via reis, zoals in:
{ t | $\exists$r $\in$ reiziger (r[faculteit] = t[faculteit] $\wedge$ $\neg\exists$ r1 $\in$ reis(r1[reisnr] = r[reisnr] $\wedge$ $\exists$a $\in$ aanvraag (a[faculteit]=r[faculteit] $\wedge$ a[reisnr] = r1[reisnr])))}
Dit levert faculteiten op met reizigers die bij een andere faculteit een aanvraag ingediend hebben.
(Er zijn overigens nog veel meer mogelijkheden om deze vraag fout te beantwoorden.)

Naam en voorletters: Ident.nr

2. Stel de volgende vraag in **twee** query formalismen naar keuze. (RA, TC of SQL)

   "Geef de naam en faculteit van reizigers, die al eens een reis hebben aangevraagd bij een andere faculteit (dan die waarvoor de reizigers werken of studeren)."

   Opmerking: je moet precies twee query talen proberen! Wie drie talen probeert krijgt 0 voor heel deze vraag. (Wie er maar 1 probeert kan ook maar 1 punt halen.)

   Antwoord:
   Wij geven het antwoord hier in de drie query formalismen. Op het tentamen was dit niet toegestaan en werd de hele vraag fout gerekend, ook als de drie uitwerkingen correct waren!

   SQL:

```
Select distinct r.naam, r.faculteit
From reiziger as r, aanvraag as a
Where r.reisnr =  a.reisnr and a.faculteit <> r.faculteit
```

   TC:

   $\{ t \mid \exists r \in \text{reiziger} (r[\text{faculteit}] = t[\text{faculteit}] \wedge r[\text{naam}] = t[\text{naam}] \wedge$
   $\exists a \in \text{aanvraag} (a[\text{reisnr}] = r[\text{reisnr}] \wedge a[\text{faculteit}] \neq r[\text{faculteit}]))$

   RA:

   $$\Pi_{r.naam,\, r.faculteit}(\sigma_{r.reisnr = a.reisnr \wedge \neg r.faculteit = a.faculteit}(\rho_r(Reiziger) \times \rho_a(Aanvraag)))$$

   Veelgemaakte fout:
   RA: join van reiziger met aanvraag: dit levert reizigers die hun eigen reisaanvraag goedkeuren.

Naam en voorletters: Ident.nr

3. Stel de volgende vraag in **twee** query formalismen naar keuze. (RA, TC of SQL)

   "Geef de naam en faculteit van de tekenbevoegde personen die al voor reizigers van alle faculteiten reisaanvragen hebben goedgekeurd."

   Opmerking: je mag ook hier slechts twee query talen proberen! Wie drie talen probeert krijgt 0 voor heel de vraag. (Wie er maar 1 probeert kan ook maar 1 punt halen.

   Antwoord:
   Wij geven het antwoord hier in de drie query formalismen. Op het tentamen was dit niet toegestaan en werd de hele vraag fout gerekend, ook als de drie uitwerkingen correct waren! Deze vraag was overigens duidelijk lastiger dan de voorgaande:

   RA:
   $(\Pi_{naam,\ faculteit,\ rfaculteit}(\sigma_{reisnr=rreisnr \wedge goedgekeurd=\text{ja}}(Aanvraag \times \rho_{r(rreisnr,\ rnaam,\ rfaculteit)}(Reiziger)))) \div \rho_{r(rfaculteit)}(\Pi_{faculteit}(Reiziger))$

   TC:
   { t | ∃a ∈ aanvraag(t[faculteit=a[faculteit] ∧ t[naam]=a[naam] ∧ ∀r ∈ reiziger(
   ∃r1 ∈ reiziger(r1[faculteit]=r[faculteit] ∧ ∃a1 ∈ aanvraag(a1[reisnr]=r1[reisnr] ∧
   a1[goedgekeurd]="ja" ∧ a1[naam]=a[naam] ∧ a1[faculteit]=a[faculteit]))))

   SQL:
   ```
   Select a.naam, a.faculteit
   from aanvraag as a
   where not exists(
      Select r.faculteit
      from reiziger as r
      where not exist(
         Select *
         From aanvraag as a1, reiziger as r2
         where a1.reisnr=r2.reisnr and a1.goedgekeurd="ja" and
             a1.naam=a.naam and a1.faculteit=a.faculteit and
             r2.faculteit=r.faculteit
   ))
   ```

   Veelgemaakte fouten:
   - slechts 1 reiziger gebruiken in TC of SQL; de all-kwantor is nodig om alle faculteiten op te noemen, de andere is nodig om een geschikte reis voor die faculteit aan te wijzen.
   - goedgekeurd <> "NULL". De beschrijving van de database geeft aan dat het attribuut NULL is tot "de uitslag voor de goedkeuring bekend is". Wanneer de aanvraag wordt afgekeurd is het attribuut niet meer NULL, maar is de reis niet goedgekeurd. (De waarde "ja" kun je overigens uit vraag 4 afleiden.)
   - goedgekeurd = "ja" bij de aanvraag die de tekenbevoegde persoon aanwijst.
   - Vergeten: a1[reisnr]=r1[reisnr]
   - .. reiziger.faculteit in (select aanvraag.faculteit from aanvraag ...)

Naam en voorletters: Ident.nr

4. Wat betekent de volgende SQL query in **normaal Nederlands**?

```
With kostprijs(reisnr, prijs) as
 (select r.reisnr, sum(t.prijs*t.aantal)
  from reis as r, reisdeel as rd, ticket as t, aanvraag as a
  where r.reisnr = rd.reisnr and rd.reisnr= t.reisnr and
  rd.reisdeelnr = t.reisdeelnr and a.reisnr = r.reisnr and
  a.goedgekeurd = "ja"
  group by reisnr)
select faculteit
from aanvraag as a, kostprijs as k
where a.reisnr = k.reisnr and
     k.prijs =(select max(k1.prijs)
               from kostprijs as k1)
```

Antwoord: Geef de faculteit die (een aanvraag voor) de duurste reis heeft goedgekeurd (d.w.z. reis met de duurste tickets).

5. Wat betekent de volgende TC query in **normaal Nederlands**?

$\{ t \mid \exists r1 \in \text{reiziger} (t[naam] = r1[naam] \wedge \exists r2 \in \text{reiziger}(r2[reisnr] = r1[reisnr] \wedge \neg(r2[faculteit] = r1[faculteit]))) \}$

Antwoord: Geef de naam van de reizigers die met een collega van een andere faculteit aan een reis deelnemen.

Opm.: aan een reis deelnemen betekent deelnemer zijn aan een reis met hetzelfde reisnr. Van deze reis is verder niets bekend. Ze kan gepland, afgerond of geannuleerd zijn.

Naam en voorletters: Ident.nr

6. Bestudeer de beschrijving van de reisadministratie-database.
   a. Identificeer de ene (en enige) functionele afhankelijkheid die geldt in deze database en die niet een sleutelafhankelijkheid is. (De sleutelafhankelijkheden die via de onderlijning van de sleutelattributen zijn gegeven mag je dus niet vermelden.) Geef bij de functionele afhankelijkheid de beschrijving uit de tekst die met die afhankelijkheid overeenkomt. Een afhankelijkheid zonder de bijpassende beschrijving of met een beschrijving die niet uit de gegeven beschrijving komt worden fout gerekend.

      Antwoord:
      Binnen de tabel aanvraag geldt "Bij elke kostenplaats(nummer) hoort één faculteit". Dit kan geformaliseerd worden tot: {kostenplaats} $\rightarrow$ {faculteit}.

   b. Je mag aannemen dat de database **niet** in BCNF is. (Anders had je vraag 6a niet kunnen beantwoorden.) Geef een lossless-join BCNF decompositie, met een beschrijving van het decompositieproces. Zonder beschrijving hoe je aan de decompositie komt wordt het antwoord fout gerekend (ook als de decompositie zelf correct is.)
      Opmerking: je decompositie wordt goedgekeurd als ze correct is uitgevoerd, ook als je ze hebt uitgevoerd met een verkeerdelijk geïdentificeerde afhankelijkheid in vraag 6a.

De tabellen Reis, Reisdeel, Reiziger en Ticket hebben alleen een sleutelafhankelijkheid en zijn dus in BCNF. De enige tabel met een functionele afhankelijkheid die geen sleutelafhankelijkheid is is de tabel Aanvraag.
Met behulp van de fd {kostenplaats} $\rightarrow$ {faculteit} kunnen we 1 stap uit het algoritme van Figuur 7.13 toepassen. We krijgen dan twee relaties:

Aanvraag1(<u>reisnr</u>, kostenplaats, naam, datum, goedgekeurd, geannuleerd) en
Kostenplaatsen(<u>kostenplaats</u>, faculteit)

In Aanvraag1 en Kostenplaatsen gelden alleen sleutelafhankelijkheden. Het algoritme van Figuur 7.13 stopt dus. De decompositie van Aanvraag in Aanvraag1 en Kostenplaatsen is een lossless-join decompositie omdat het gebruikte algoritme dat garandeert. (Zie Silberschatz pag. 281: "The decomposition that the algorithm generates is not only in BCNF, but is also a lossless-join decomposition."
Je kunt het argument ook expliciet opschrijven: Aanvraag1 $\cap$ Kostenplaatsen = {kostenplaats} en {kostenplaats} is een superkey van Kostenplaatsen, zodat de voldoende voorwaarde voor lossless-join decompositie, beschreven op pag. 276 van Silberschatz is voldaan, namelijk: Aanvraag1 $\cap$ Kostenplaatsen $\rightarrow$ Kostenplaatsen, of met andere woorden {kostenplaats} $\rightarrow$ {kostenplaats, faculteit}.
De decompositie is tevens dependency preserving (maar dat werd niet gevraagd).

Veelgemaakte fout:
- Geen argumentatie voor het in BCNF zijn van het antwoord. (wat zijn de nieuwe keys, of verwijzing naar eigenschap van Algoritme 7.13)
- Geen argumentatie voor de loss-less join eigenschape (kostenplaats is superkey van Kostenplaatsen, of eigenschap van Algoritme 7.13).

Naam en voorletters: Ident.nr

7. De universiteit wil sommige eigenschappen van de reisadministratie veranderen. De volgende eigenschappen worden toegevoegd of veranderd, en leiden tot veranderingen aan het database schema.
   a. De universiteit wil geen onderscheid meer maken tussen "vertrekpunt" en "startplaats" en tussen "aankomstpunt" en "bestemming" van reisdelen en tickets. Zorg ervoor dat de redundantie die hierdoor ontstaat (omdat vertrekpunt en startplaats hetzelfde zijn geworden, en aankomstpunt en bestemming ook) niet meer voor komt. Leg uit waarom (op basis van de beschrijving van de database!) de verandering die je voorstelt is toegestaan. Uitleg die niet gerelateerd is aan de beschrijving van de database wordt fout gerekend!

      Antwoord: de tabel die hierdoor geraakt wordt is Ticket. In eerste instantie betekent het wegvallen van het onderscheid, dat er twee attributen in Ticket hernoemd worden:
      Ticket(reisnr, reisdeelnr, datum, startplaats, vertrektijd, bestemming, aankomsttijd, vervoerder, aantal, prijs). Merk op dat we ook een attribuut in Ticket en een attribuut in Reisdeel hadden kunnen hernoemen of twee attributen in Reisdeel.
      Er geldt: "Elk ticket moet overeenkomen met een reisdeel (van dezelfde reis) maar er mogen reisdelen zijn zonder ticket.", ofwel
      $\Pi_{reisnr, reisdeelnr}(Ticket) \subseteq \Pi_{reisnr, reisdeelnr}(Reisdeel)$
      Verder geldt na de hernoeming in zowel Ticket als Reisdeel:
      {reisnr, reisdeelnr} $\rightarrow$ {startplaats, bestemming}
      Op grond hiervan kunnen we Ticket opsplitsen in twee tabellen (lossless-join in BCNF):
      Ticket1(reisnr, reisdeelnr, datum, vertrektijd, aankomsttijd, vervoerder, aantal, prijs)
      Routedeel(reisnr, reisdeelnr, startplaats, bestemming).
      Noem TK = $\Pi_{reisnr, reisdeelnr}(Ticket)$ =
      $\Pi_{reisnr, reisdeelnr}(Ticket1)$ = $\Pi_{reisnr, reisdeelnr}(Routedeel)$
      Vanwege de afhankelijkheden geldt nu:
      $Routedeel = \Pi_{reisnr, reisdeelnr, startpunt, bestemming}(Reisdeel \rhd\lhd TK)$ zodat Routedeel kan vervallen zonder verlies aan informatie.

      Merk op, dat vanwege het feit dat er niet voor elk reisdeel een ticket hoeft te zijn, de twee attributen niet uit Reisdeel weggelaten kunnen worden. Dit zou wel leiden tot informatieverlies.

      Veelgemaakte fouten:
      - bestemming van reis erbij betrekken.
      - verband leggen tussen ticket en reisdeel door alleen reisdeelnr.
      - alleen resultaat geven, geen argumentatie.

Naam en voorletters: Ident.nr

b. Er mogen in een reis geen twee reisdelen zijn met dezelfde vertrekdatum en startplaats. Beschrijf deze eigenschap als functionele afhankelijkheid op de tabel Reisdeel. Zorg ervoor dat deze tabel met de nieuwe beperking erbij in 3NF is. (Je hoeft dus alleen maar naar de tabel Reisdeel te kijken. Negeer alle andere tabellen.) Indien nodig moet je een decompositie uitvoeren, maar die moet dependency-preserving zijn. Het in 3NF zijn en/of het mogelijk decomponeren moet van de nodige uitleg worden voorzien. Zoniet wordt zelfs een correct schema in 3NF als een fout antwoord gerekend.

Binnen een reis moet dus gelden, dat vertrekdag en startplaats het reisdeelnummer uniek bepalen. Dit kan geformuleerd worden als:

{reisnr, vertrekdag, startplaats} -> {reisnr, reisdeelnr}.

Dit kunnen we vereenvoudigen tot {reisnr, vertrekdag, startplaats} -> {reisdeelnr}.
De rechterkant van de FD is een primair attribuut, want onderdeel van de key van Reisdeel.
We kunnen dus concluderen, dat Reisdeel in 3NF is met de geformuleerde afhankelijkheid.
Merk op, dat $\{reisnr, vertrekdag, startplaats\}^+$ = Reisdeel. De tabel is dus ook in BCNF.

Veelgemaakte fouten:

- {vertrekdag, startplaats} $\rightarrow$ {reisnr, reisdeelnr}
- {reisnr, reisdeelnr} $\rightarrow$ {vertrekdag, startplaats}
- tabel splitsen met een niet-dependency preserving of niet lossless-join resultaat
- {reisnr, reisdeelnr} superkey vervangen door {reisnr, vertrekdag, startplaats} superkey. Dat mag maar is nog geen uitleg voor het bewaren van 3NF.

Naam en voorletters: Ident.nr

8. Neem een relatieschema R en laat V, W, X, Y, Z verzamelingen attributen zijn.
   R1: Strikte Reflectiviteitsregel: Er geldt altijd dat $X \rightarrow X$.
   R2: Augmented-Union regel: Als $X \rightarrow Y$ en $V \rightarrow W$, dan $XV \rightarrow YW$.
   R3: Transitiviteit: Als $X \rightarrow Y$ en $Y \rightarrow Z$, dan $X \rightarrow Z$.

   Bewijs, dat R1, R2 en R3 correct zijn. Bewijs ook dat R1, R2 en R3 samen geen volledig stel afleidingsregels zijn (voor fd's). Hint: kijk naar bewijzen van niet-redundantie!

   Bij deze vraag mag je aannemen, dat F1, F2 en F3 (reflexivity, augmentation, transitivity, pag 265 in Silberschatz) correct en volledig zijn (maar je hoeft dit niet te gebruiken als je niet wil).

   Antwoord: R1, R2 en R3 zijn correct wanneer ze afleidbaar zijn uit (de correcte regels) F1, F2 en F3.

   - R3 is identiek aan F3 en dus correct.
   - Omdat $X \subseteq X$, geldt met behulp van F1, dat $X \rightarrow X$, dus R1 volgt uit F1 en is dus correct.
   - Met behulp van F2 volgt uit $X \rightarrow Y$, dat $XV \rightarrow YV$ en uit $V \rightarrow W$, dat $YV \rightarrow YW$. Uit deze twee resultaten volgt met transistiviteit, dat $XV \rightarrow YW$, ofwel, R2 volgt uit F2 en F3 en is dus correct.

   Het stelsel R1, R2 en R3 is niet volledig. Beschouw het volgende voorbeeld:

   $R = \{A\}$, $F = \emptyset$, dan geldt: $F_R^+ = \{ A \rightarrow A, \emptyset \rightarrow \emptyset \}$, waarbij het subscript R aangeeft dat R1, R2 en R3 gebruikt zijn om F^+ te berekenen.

   Echter, onder gebruik van de regels F1, F2 en F3 krijgen we: $F_F^+ = \{ A \rightarrow A, A \rightarrow \emptyset , \emptyset \rightarrow \emptyset \}$.

   Veelgemaakte fouten

   - alleen analyse geven, geen tegenvoorbeeld bij b). een bewering als “ik kan F1 niet afleiden uit R1, R2 en R3” is geen bewijs.
   - onvolledig tegenvoorbeeld geven, door bijvoorbeeld alleen een afhankelijkheid te noemen, zonder R of F te vermelden.

Waardering:

Als je geen vrijstelling hebt voor het pretentamen moet je dat eerst doen, en als je de 3 vraagjes niet alle drie goed hebt krijg je het cijfer 1.
Als je vrijstelling hebt of het pretentamen hebt gehaald is de waardering als volgt:
pretentamen vragen: 0 punten, hoofdtentamen vraag 1: 3 punten (1 punt per query taal), vraag 2: 2 punten (1 punt per query taal), vraag 3: 2 punten (1 punt per query taal), 4 en 5: elk 1 punt, vraag 6a: 1 punt, 6b: 1 punt, vraag 7a: 1 punt, 7b: 1 punt, vraag 8: correct 1 punt, niet-volledig 1 punt. Elke vraag levert 1 heel punt op of 0 punten. Er bestaat geen “half correct” antwoord. De cijfers tellen op tot 15. Deze worden als volgt herleid tot een cijfer op 10:
De cijfers voor de query vragen worden opgeteld (maximum is hier 9). Elk van die punten wordt gedeeld door 2. (Je haalt hier dus 0 tot 4.5 mee.) Dit is het query-deelcijfer.
De cijfers voor de design vragen worden opgeteld (maximum is hier 6). Elk van die punten wordt ook gedeeld door 2. (Je haalt hier dus 0 tot 3 mee.)
Het totaal te behalen punten op deze manier is dus 7.5 (4.5 + 3).
Als je voor het query deel minstens 2.5 en voor het design deel minstens 1.5 hebt gehaald krijg je een bonuspunt kado (dit is beloning voor evenwichtige kennis over de beide delen van het college). Daarnaast krijgt iedereen een punt kado voor deelname. Het totaal aantal te behalen punten is dus 9.5. Wanneer het behaalde cijfer 5.5 is of meer, en geen geheel getal, dan wordt het naar boven afgerond.
Wanneer het getal 4.5 is of minder, en geen geheel getal, dan wordt het naar beneden afgerond.