Relationele gegevensbanken: examens

Examen 2016-2017 (eerste zit)

**Eerste deel (voormiddag):**

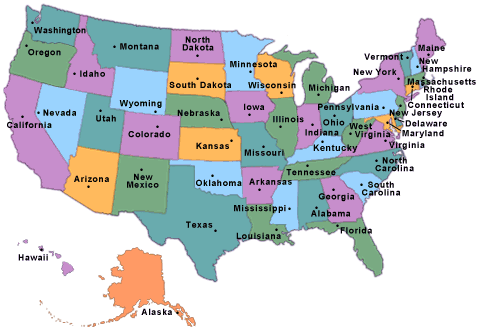
Theorievragen. (Als je ze nog weet, of als je ze op klad neergeschreven hebt, zou je die dan hier graag willen zetten?)

**Tweede deel (voormiddag):**

4 opdrachten die in totaal voor de helft van de punten van de oefening gedeelte van het examen meetellen.

Oefening 1:

Je moest a.d.h.v. de tabellen ‘regios’ en ‘grenzen’ voor de VSA tripletten van staten vormen. Staat X heeft een grens met staat y, staat Y heeft een grens met staat Z en staat Z heeft een grens met staat X.



Zo is ‘Idaho - Oregon - Washington’ een triplet, of bv. ‘California - Nevada - Arizona’.

De resultaattabel moest eveneens naast 3 kolommen voor de namen van de staten X,Y en Z ook een kolom bevatten die de gemiddelde longitude van deze 3 staten weergeeft. De tabel moest dan geordend worden op deze gem. longitude (asc).

Opmerking:

* De tripletten moesten wel uniek zijn. Bijgevolg is een tabel met bv. ‘California - Nevada - Arizona’ én ‘California - Arizona - Nevada’ én ‘Arizona - California - Nevada’ foutief.
* CTE’s en subqueries waren niet toegelaten. Ik had het opgelost d.m.v. meerdere equi- en auto-joins.
* Makkelijkste opdracht (vond ik persoonlijk)

Oefening 2:

Oef. waarbij je de top 3, top 7, … moet bepalen in blokken gegevens. Er waren 7 kolommen: eerste kolom stond voor de discipline, 2de en 3de vormden samen één blok die informatie toonden over de top 3 competitors. Er waren 6 rijen voor deze blok: de eerste 3 voor de top -3 mannelijke competitors en de laatste 3 voor de top - 3 vrouwelijke competitors waarbij de 2de kolom de naam weergaf van de competitor en de 3de het aantal overwinningen. De top competitors worden geselecteerd volgens het aantal overwinningen (ik had count(case when rank=1 then 1 end) gebruikt).

Analoog zorgden de 4de en 5de kolommen voor informatie ivm de top 7 competitors en de 6de en 7 voor de top - 12 (ik ben niet zeker over de laatste, kon ook top 14 geweest zijn, maar het maakt niet echt uit).  
  
De eerste “rij” blokken waren voor één discipline bedoelt, ‘KB’ denk ik. Eronder moest er een andere “rij” blokken voor de volgende discipline enz.

Je mowcht geen top functie gebruiken, subqueries waren niet toegelaten. Aggregaat en de bijhorende rapporteringsfuncties waren wel toegelaten, alsook de ranking functies. Dus je moest de top-functie simuleren met een CTE of een autojoin.  
  
Oefening 3:

results en races gebruiken. Per skiër alle wedstrijden bepalen waarin hij een podiumplaats heeft behaald maar geen overwinning tussen 2 opeenvolgende wedstrijden waarin hij de eerste plaats behaald heeft. Zowel voor de mannen als voor de vrouwen apart diegene overhouden met de meeste wedstrijden. naam alleen tonen op eerste rij. disciplines doen er niet toe.

zoieets van output bekomen

name racedate discipline podiumplaats

man dd/mm/yyyy a 1

dd/mm/yyyy b 2

dd/mm/yyyy a 2

dd/mm/yyyy c 3

dd/mm/yyyy d 2

……….. ….. ……..

dd/mm/yyyy b 1

vrouw dd/mm/yyyy a 1

dd/mm/yyyy b 2

dd/mm/yyyy g 2

dd/mm/yyyy c 3

dd/mm/yyyy d 2

……….. ….. ……..

dd/mm/yyyy e 1

Oefening 4:

results en races tabel gebruiken. per competitor bepalen wat zijn langst mogelijke streak is van opeenvolgende seizoenen waarin hij elk seizoen minstens een overwinning behaald heeft. Er mag in elke lijst van opeenvolgende seizoenen echter 1 seizoen ontbreken. ontbrekend seizoen ook geven. alleen de 10 competitors overhouden met de grootste streak.

Bv: BEGIN EINDE ontbrekend seizoen

goetsch renate 1993 2007 1996

**Derde deel (namiddag):**

Enig te raadplegen ‘tabel’: dual

Opmerkingen i.v.m. de quotering:

* De aanbevolgen werkwijze voor de oplossing (zie onderaan bestaat uit 2, nagenoeg onafhankelijke, opeenvolgende stappen
* Indien je de eerste stap niet correct kan implementeren, dan kan je deze eventueel bypassen door gebruik te maken van (select \* from partities) om de 2de stap te kunnen realiseren

Beperkingen:

* Opdracht mag enkel uit één enkele SQL querie bestaan, eventueel voorafgegaan door een CTE, geen PL/SQL
* Samengestelde gert. operaties (union, union all, ... ) verboden
* Hoewel het niet absoluut nodig is, zijn gecoreleerde subqueries toegelaten
* Enkel standaard beschikbare vormen van analytische functies zijn toegelaten: van aggregaatfuncties afgeleide reportingfuncties, rankingfuncties
* Geen gebruik van pseudokolommen zoals rouwnum of rowid
* Gebruik van hiërarchische queries is verplicht

Een partitie van een getal is elke verzameling gehele getallen die als som het getal opleveren. Bijzeonder interessant zijn partities waarbij de termen bestaan uit onderling verschilllende getallen. Bepaal via SQL alle mogelijke dergelijke partities van een specifiek getal die gehardcodeert mag worden, bij voorkeur slechts één keer.

Twee partities die door permutaties van de termen in elkaar kunnen overgaan, worden hierbij niet als verschillend beschouwd. Sortering van de termen is bijgevolg ook noodzakelijk.

Ook de diverse partities zelf moeten gesorteerd worden, in de sorteervolgorde van de opeenvolgende kleinste termen (eerst op de kleinste term, vervolgens op de één na kleinste term, …)

Voorbeeld:

G PARTITIE

------------ ------------------

6 1 + 2 + 3

6 1 + 5  
 6 2 + 4

6 6

Je moet de sorteervolgorde van deze voorbeelden exact reproduceren. Zo geldt er bv. in de partitie van 15: …….. < 1 + 2 + 5 + 5 + 7 < 1 + 2 + 12 < 1 + 3 + 4 + 7 < …………

Aanbevolen werkwijze:

1. Genereer via een hiërarchische query een uitvoertabel die mogelijke kandidaatpartities (met termen bestaand uit onderling verschillende getallen) produceert van een specifiek getal. De uiteindelijke gezochte partities zullen een deelverzameling vormen van deze verzameling kandidaatpartities  
     
   De in deze stap geconstrueerde tabel moet voor elke kandidaatpartitie minstens een stringveld vertonen, die in de verschillende termen van de kandidaatpartitie aan elkaar geconcateneerd zijn, met een ‘ + ‘ -teken als delimiter. De diverse termen moeten hierbij reeds correct gesorteerd zijn. Er moet hier echter nog niet gecontroleerd worden of dat de som v.d. termen van de kandidaatpartitie inderdaad het specifieke getal oplevert. Indien je deze stap niet correct zou kunnen implementeren, dan kan je eventueel het bypassen door gebruik te maken van de tabel ‘partities’. Deze bevat alle partities van een willekeurige getal <= 40, in het hierboven vermelde kandidaat.
2. Selecteer, hetzij uit de in stap 1 gegenereerde uitvoertabel, hetzij uit de extra partitie tabel, enkel die kandidaatpartities waarvan de som van de termen het opgegeven specifieke getal oplevert.  
     
   Je zal hierbij de diverse termen van de geconcateneerde stringveld moeten extraheren, om er de som van de kunnen berekenen. Het correct sorteren van de partities onderling kan volledig in deze stap gebeuren, maar kan eventueel ook wat voorbereid worden in stap 1

Hou er, om het aantal iteraties in de hiërarchische (recursieve) query van stap 1 te beperken, rekening mee dat de som van 2 opeenvolgende termen van een partitie niet groter kan zijn dan het opgegeven getal. Deze vaststelling is zeer eenvoudig te implementeren en is cruciaal om de uitvoeringstijd te beperken.

Om het aantal iteraties nog meer te minimaliseren, voor meer performantieverbetering, moet je ermee rekening houden dat voor het aantal termen, waaruit een partitie in van elkaar verschillende termen bestaan, a priori een bovengrens gekend is:

* De partities van alle getallen kleiner dan N\*(N+1)/2 kunnen maximaal slechts uit N - 1 verschillende termen bestaan. Bv. 9 (< 4\*5/2) heeft partities met elk maximaal 3 termen, heeft 12 (< 6\*7/2) partities met max. 4 termen
* Getallen van de vorm N\*(N+1)/2 hebben, behalve één enkele partitie die uit N-1 verschillende termen bestaat, precies één enkele partitie die uit N verschillende termen bestaat, namelijk de verzameling van alle opeenvolgende gehele getallen tot N: zie bv. partitie 6 = 1 + 2 + 3 of 15 = 1 + 2 + 3 +4 + 5

Examen 2014-2015 (eerste zit)

20 theorievragen. Volgens mij dezelfde als vorig jaar.

Voormiddag; 4 u tijd (minus de tijd die je nodig hebt voor de theorie)

4 query's, nergens recursie of scalaire subquery's

Namiddag: 3u tijd

3 query's (2 met recursie)

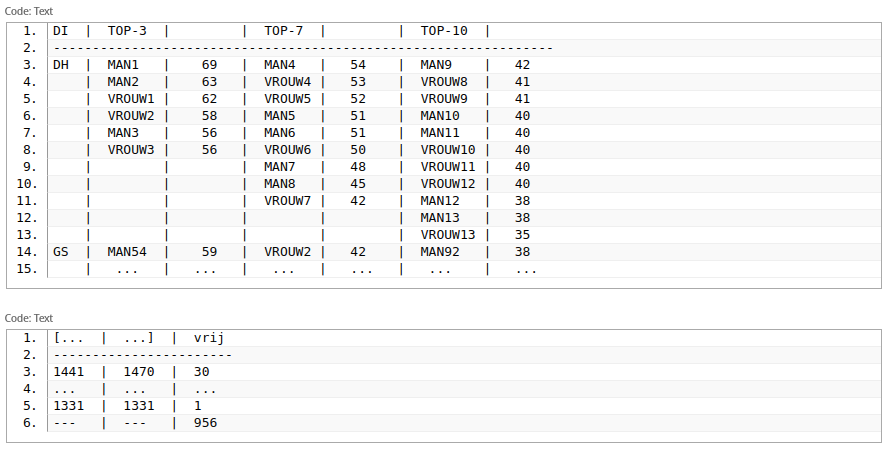
Oefeningen:

**Oefening 1**

Deze is mij ontgaan, sorry.

**Oefening 2**  
Geef per discipline een in gender gemengde top-3, top-7 en top-10

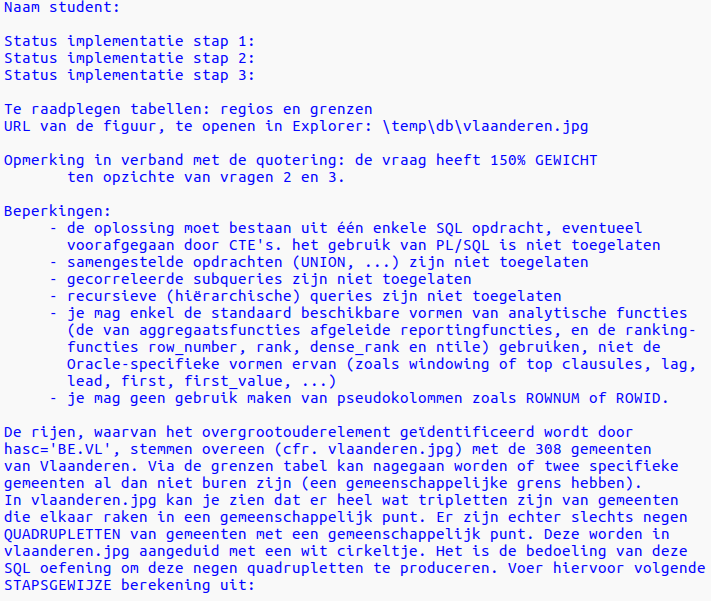
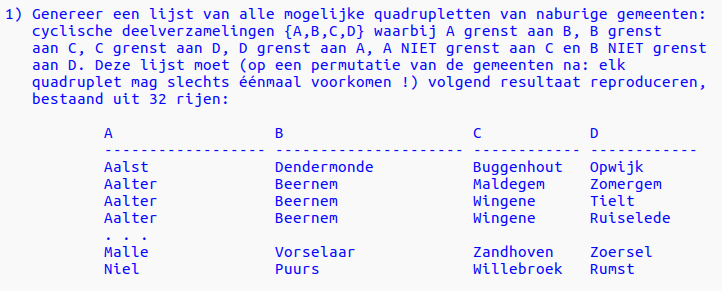
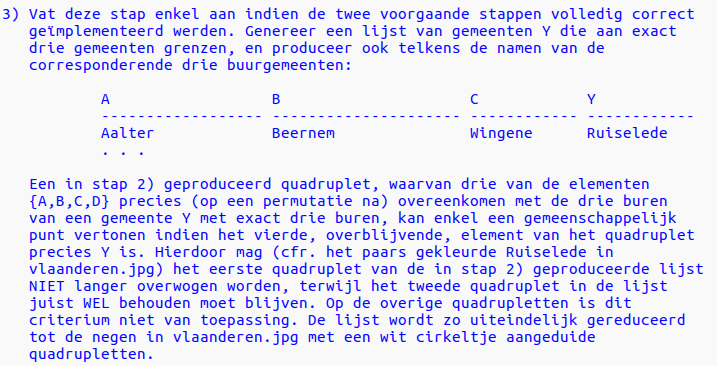
op basis van het aantal overwinningen die de skiër/skiester in die discipline behaalde. Doe dit in blokken (zie voorbeeld uitvoer). Zowel de top-3 van de mannen als de top-3 van de vrouwen moet in het top-3 blok aanwezig zijn (zie voorbeeld uitvoer). Herhaal de top-3 en top-7 niet in de 'lagere' blokken. Zorg dat alles zo compact mogelijk wordt weergegeven (= geen lege lijnen). Toon voor elke persoon het aantal overwinningen. Orden op dit aantal en nadien op de naam van de persoon.

Michael :

**Oefening 3**

Bijna elke skiër/skiester haalt tussen twee opeenvolgende overwinningen enkele keren het podium als tweede of derde. Zoek de skiër en skiester die tussen twee overwinningen het meeste op het podium stonden (rank=2 en rank=3). Voor deze periode geef je:

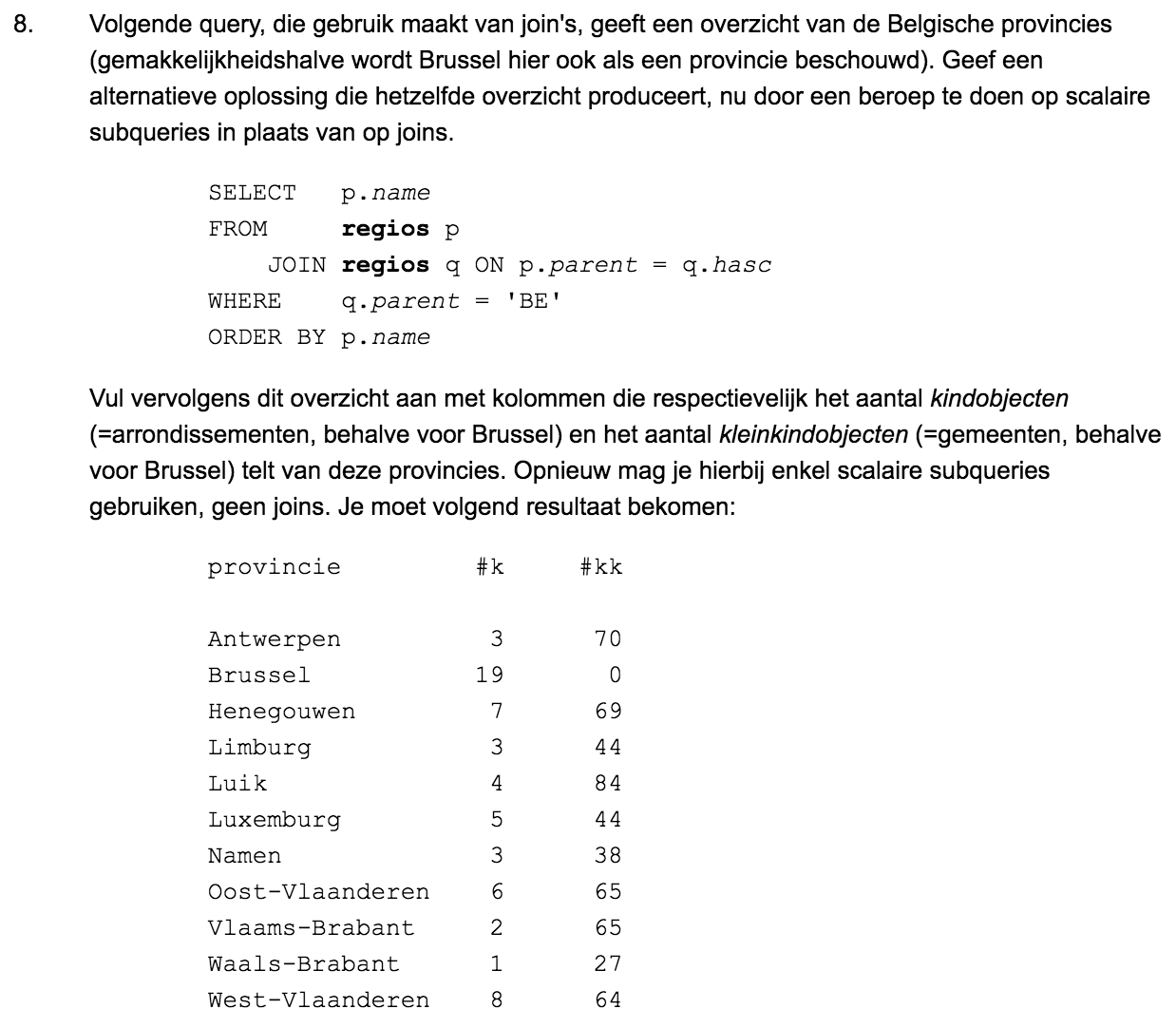
1. De naam van de persoon, maar dit enkel op de eerste rij.
2. *Racedate*, plaats (resort *name*), *discipline* en *rank* van de podiumplaats.
3. (Inclusief de overwinningen die de periode afbakenen)

**Oefening 4**  
  
  
  
  


**Oefening 5**

Bij deze vraag mocht je **geen gebruik** maken van **CTE**'s, **JOIN’**s, analytische functies met **OVER()** extensie! Je moest gebruik maken van scalaire subquery's. (Stuur mij een bericht als de gebruikte termen verkeerd zijn. Ik sla sommige begrippen door elkaar.). Om te beginnen moest je een gegeven query met een JOIN omvormen tot een query die enkel gebruikt maakte van scalaire subquery's. Deze moest natuurlijk hetzelfde resultaat geven; namelijk alle landen van de EU (gebruikmakend van de *members* en *cities* tabel). Verder moest je van die landen het aantal kinderen, kleinkinderen en achterkleinkinderen geven.

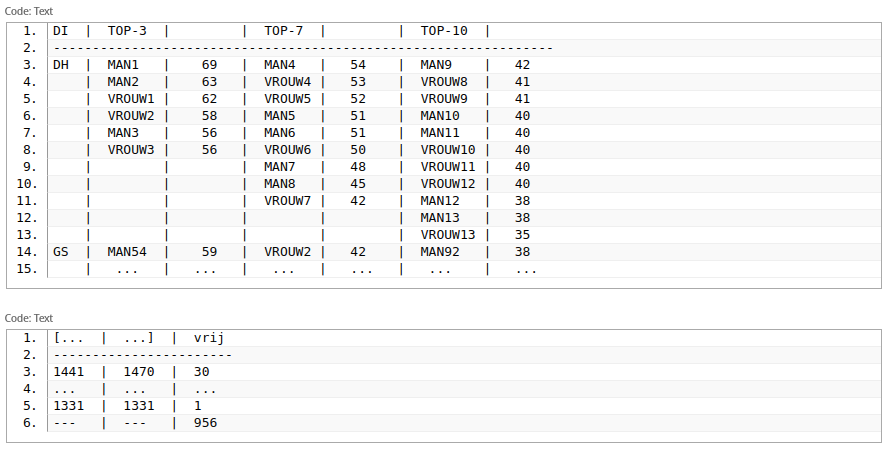
[Gelijkaardig aan oefening 8 van reeks 7?](https://intranet.tiwi.ugent.be/Databanken/Labo/07/07.sql)



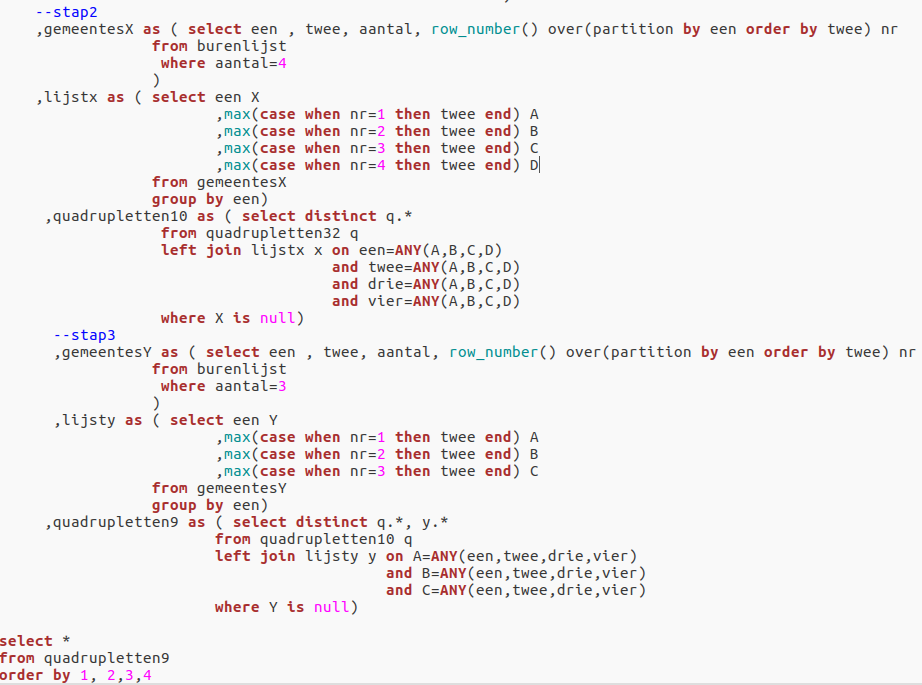
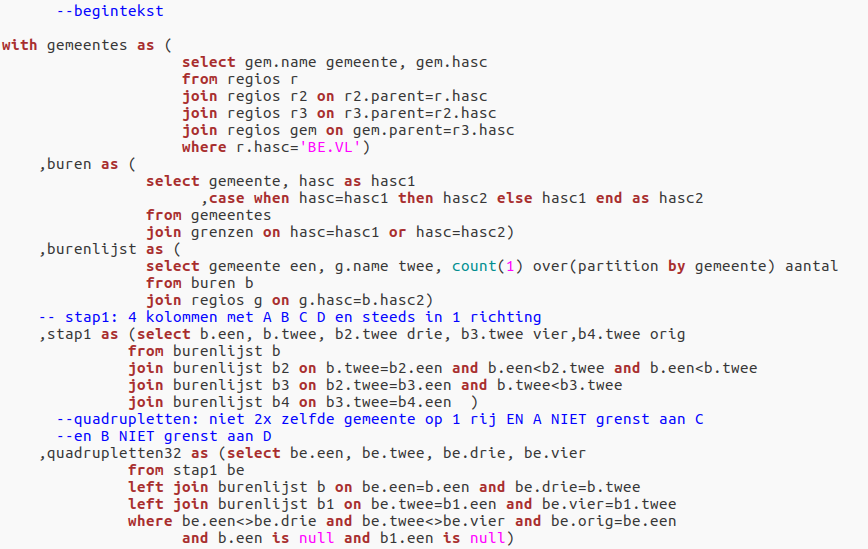
**Oefening 6**

Een oefening zoals gezien in de theorieles (althans vorig jaar) op het vinden van een reisweg. Er zijn twee CTE's gegeven:   
  
- **landen**: alle nodige info (naam, hasc) van de landen van Zuid-Amerika  
- **buren**: alle mogelijke buren van elk land uit Zuid-Amerika (hasc1, hasc2) in beide richtingen (dus '*dubbels*') om het makkelijker te maken  
  
Je start en eindigt in Brazilië (hasc='BR') en je moet een zo lang mogelijke reis maken. Je mag elk land maar één keer bezoeken. De reisweg ziet er **[zo](http://i.snag.gy/xDvtB.jpg)** uit. Je moest als resultaat twee lijnen bekomen (het rode pad in tegen- en wijzerzin).

**Oefening 7**

Er was een CTE **schouwburg** geconstrueerd uit de *elevation*s van de Cities tabel. Deze CTE moest je zien als de stoelnummers in een zaal die reeds bezet waren. De zaal had 60 rijen met elk 30 stoelen (stoelnummers [1,1800]). Je moest zo beknopt mogelijk PER RIJ de lege plaatsen tonen. Dit wou dus zeggen dat een interval een maximumgrootte van 30 had. Je moest ordenen op het aantal vrije plaatsen, maar onderaan de tabel moest het *totaal aantal* lege plaatsen staan ter controle. Voor een voorbeeld van de uitvoer, zie hier onder:   
  


**Oplossing oefening 4**



Examen 2013-2014 (tweede zit)

Examen bestond weer uit drie delen.

* **Deel 1**
* Ongeveer 20 theorievragen; goed voor 25% van de punten.
* **Deel 2**
* Vijf query's opstellen. Geen UNION, ... geen recursieve functies gebruiken, basis analytische functies toegestaan, CTE's toegestaan.
* Eerste drie vragen samen goed voor 25% van de punten, laatste twee samen ook goed voor 25% van de punten.
* **Deel 3**
* Twee recursieve query's schrijven; ook weer goed voor 25% van de punten.

Theorie:

De vragen gingen onder meer over: heap-bestanden, geclusterde index, CTE's vs views, with check option, wanneer 3GL, granulariteit van de lock, cursors.

Details van de vragen ben ik kwijt, maar indien die begrippen goed gekend zijn is al meer dan de helft opgelost.

Oefeningen (niet-recursief):

Heb ik niet echt onthouden. Enkel de laatste vraag kan ik me nog deels herinneren.

Bij vraag 5 had je de tabellen Regio's en Grenzen nodig en moest je in drie stappen te werk gaan. Het doel van de vraag was om 4 dorpen die elkaar in één punt ontmoeten. Je zou dus je tegelijkertijd rechterhand in dorp1, je linkerhand in dorp2, je rechtervoet in dorp3 en je linkervoet in dorp4 kunnen leggen/zetten.

Na de eerste stap was het de bedoeling dat je 32 rijen overhield. Dorp1 moest grenzen aan dorp2, wat moest grenzen aan dorp3, wat op zijn beurt weer moest grenzen aan dorp4, wat ook grensde aan dorp1. Dorp1 en dorp3, alsook dorp2 en dorp4 mochten niet grenzen.

Uiteindelijk, na de derde stap, mocht je maar 8 rijen overhouden.

Er waren ook vragen met cumulatieve sommen, vragen waar je bij het groepen het seizoen en geslacht maar 1 keer mocht tonen, enz..

Iemand anders:

Vrij belangrijk: gecorreleerde subqueries waren niet toegestaan

Vraag 1: leek heel hard op oef 23 van labo 6

Vraag 4: De mondialisering in het skiën wordt als volgt gemeten: voor elke wedstrijd bepaal je de grootst mogelijke M waarvoor geldt dat de nationaliteiten van de top-M allemaal verschillend zijn (tabellen: races, results, competitors). Produceer als uitvoer alle wedstrijden waarvoor M maximaal 2 verschilt van de grootste M over alle wedstrijden (die is trouwens 10, maar dat mag je niet hardcoderen), in dit formaat:

Mammoth Mountain 7/03/1997 L SL 1 WIBERG Pernilla SWE

2 EGGER Sabine AUT

3 MAGONI Lara ITA

4 ROTEN MEIER Karin SUI

5 EDER Elfi GRN

6 COMPAGNONI Deborah ITA

7 WACHTER Anita AUT

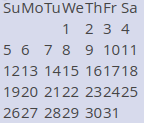
8 ACCOLA Martina SUI

9 SALVENMOSER Ingrid AUT

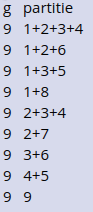
Voor elke wedstrijd die voldoet, schrijf je dus de top-M uit. Het resort, de datum, het geslacht en de discipline worden enkel op de eerste rij weergegeven. (Deze wedstrijd heb ik compleet willekeurig gekozen trouwens, M is hier 5). Sorteer op dalende M en vervolgens op datum.

Oefeningen (recursief):

Bij vraag 1 (6) moest je een kalender van deze maand maken. Niets hardcoderen:



Vraag 2 (7) ging over 'partities'. Bedoeling was om voor een bepaald getal (bijvoorbeeld 9) volgende uitvoer te krijgen:



Er stonden geloof ik ook gelijkheidstekens tussen. Het MOEST in deze volgorde! Kleinste waarden meest links (1+2+3+4 en niet 3+2+4+1 ofzo), en 1+2+6 is 'kleiner' (lees: staat hoger) dan 1+3+5 want 2 is kleiner dan 3.

Examen 2013-2014 (tweede zit)

Theorie:

20 theorievragen zoals in januari (Spoiler: Het waren dezelfde vragen)

Oefeningen:

6 oefeningen, gecorreleerde subqueries nooit toegestaan

**Oefening 1**

Uit results en races voor alle competitors zoeken wat de maximale tijd was tussen 2 opeenvolgende overwinningen, daar dan de top 20 uit halen. Vervolgens het totaal aantal overwinningen voor elke competitor bepalen voor enkel die disciplines van die twee opeenvolgende overwinningen. Analytische functies toegestaan, geen recursieve CTE's.

**Oefening 2**

Uit results en races de langste periodes (een periode is een reeks van opeenvolgende jaren) zoeken waarbij een competitor elk opeenvolgend jaar in die periode minstens één overwinning heeft gehaald, vervolgens filteren op een minimum van 5 jaar. Dit was m.a.w een oefening op intervallen maar i.p.v op een reeks elevations zoals gezien in de labo's is het ditmaal op een reeks seizoenen waarin een persoon minstens één overwinning heeft gehaald. Analytische functies waren toegestaan, waardoor meerdere oplossingen mogelijk waren. Ikzelf koos voor de oplossing met de dense\_rank functie.

**Oefening 3**

Eerst alle provincies en hun bijhorende gemeentes + populaties geven. Vervolgens de cumulatieve populaties voor elke provincie en zijn gemeentes geven in volgorde van hoogste naar laagste populatie. Daarna filteren zodanig dat men enkel de grootste gemeentes per provincie overhoudt wiens cumulatieve populatie net meer dan 1/3de van de totale populatie van die provincie bedraagt. Niet zeker of analytische functies toegelaten waren hier, er was slechts één oefening waarbij ze verboden waren. Ik denk dat het deze was?

**Oefening 4**

Oefening op medianen. Voor elke provincie werden al zijn meetpunten (elevation) gegeven als CTE. Gevraagd was om de mediaan, het eerste kwartiel (idem als mediaan maar voor 25%) en het derde kwartiel (idem als mediaan maar voor 75%) te geven voor elk van de provincies. Oftewel waren analytische functies bij de vorige oefening verboden, oftewel bij deze oefening.

--Einde niet-recursieve oefeningen

**Oefening 5**

Gepivoteerde tabel opstellen voor alle competitors tussen seizoenen 2006 en 2010 met de totale behaalde punten voor elk van de disciplines per seizoen. De totaalsom is i.p.v. zoals gezien in de oefeningen nu niet een werkelijke wiskundige som maar een string van de vorm "80+80+70+45+30" (van groot naar klein vereist). Analytische functies + recursieve CTE's toegelaten.

**Oefening 6**

De zwaarste oefening, op recursie natuurlijk. De opgave is me jammer genoeg onbekend. Ik weet alleen dat hij voor 50% meer meetelde dan de andere oefeningen, maar het was bijna tijd en de goesting was verdwenen. Misschien is er nog een verdwaalde ziel op dit forum die wat kan zeggen over de opgave.

Examen 2006-2007 (tweede zit)

Theorie

Hoodstuk 6:

* Constraints uitleg + welke soorten constraints? + hoe? wat code?
* Verschil tussen view en CTE, welke eigenschappen, gelijkenissen,uitbreidingen...
* Verschillende varianten van bestandsorganisatie bij indexering geven

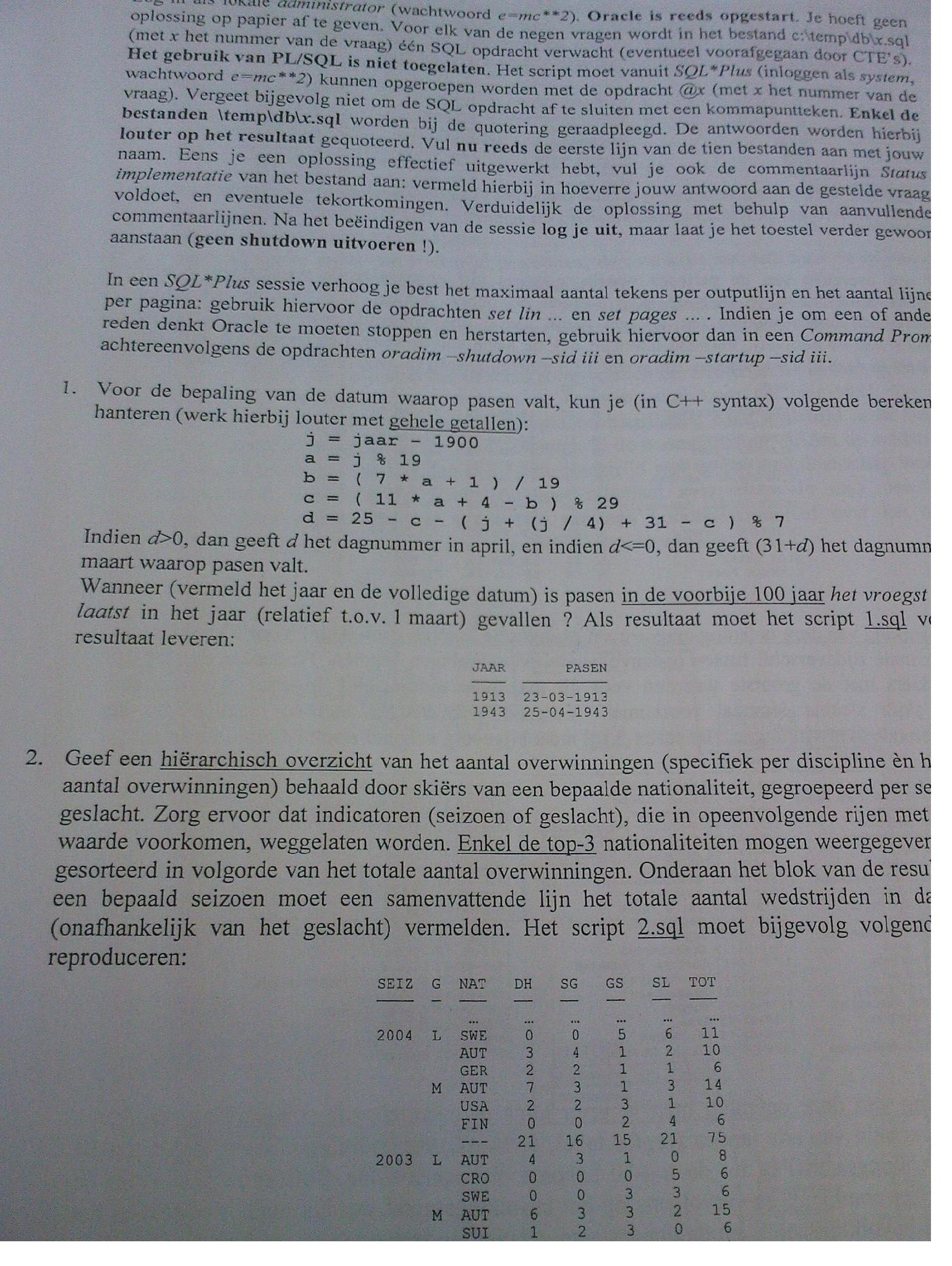
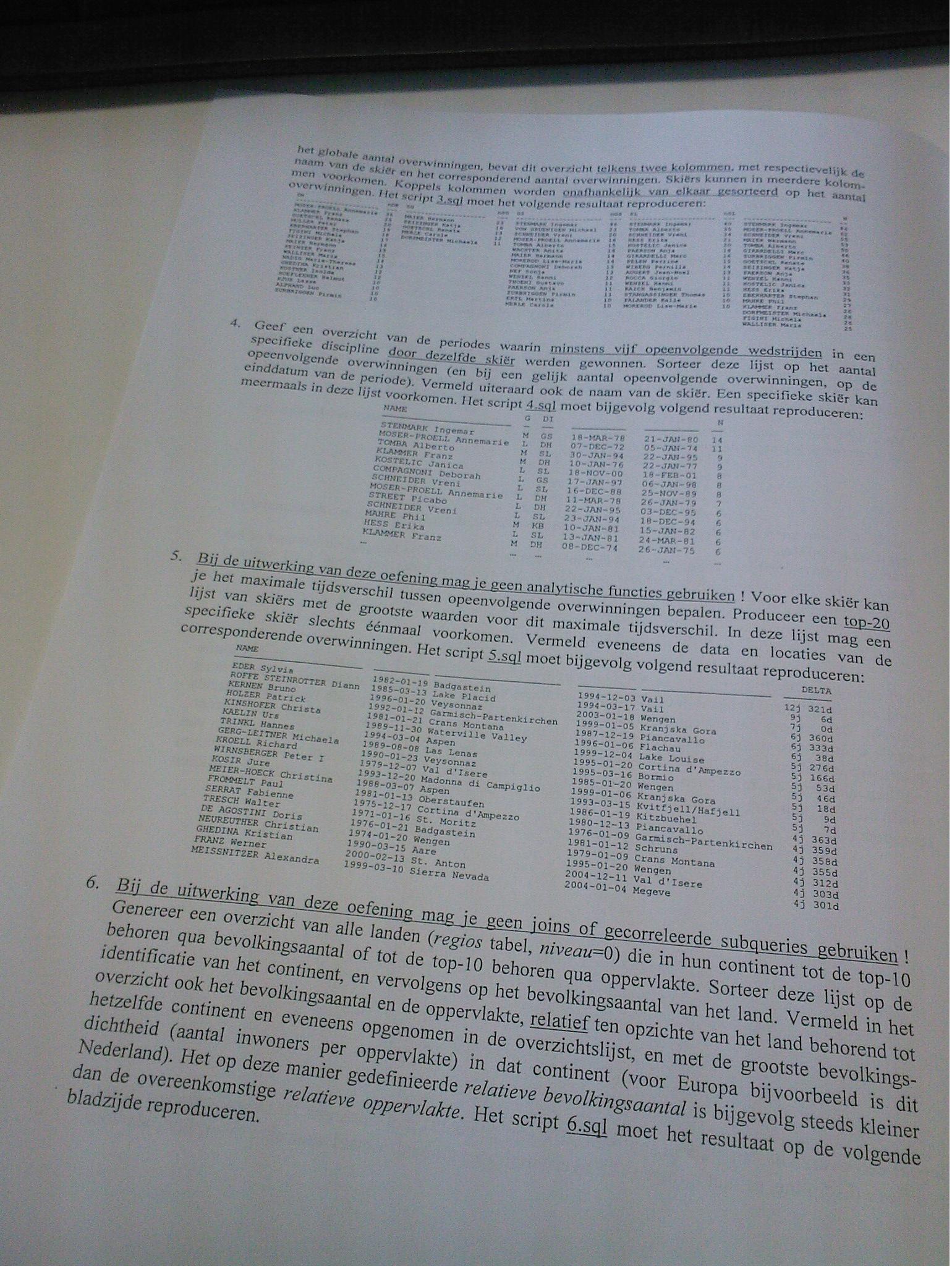
Hoofdstuk 7:

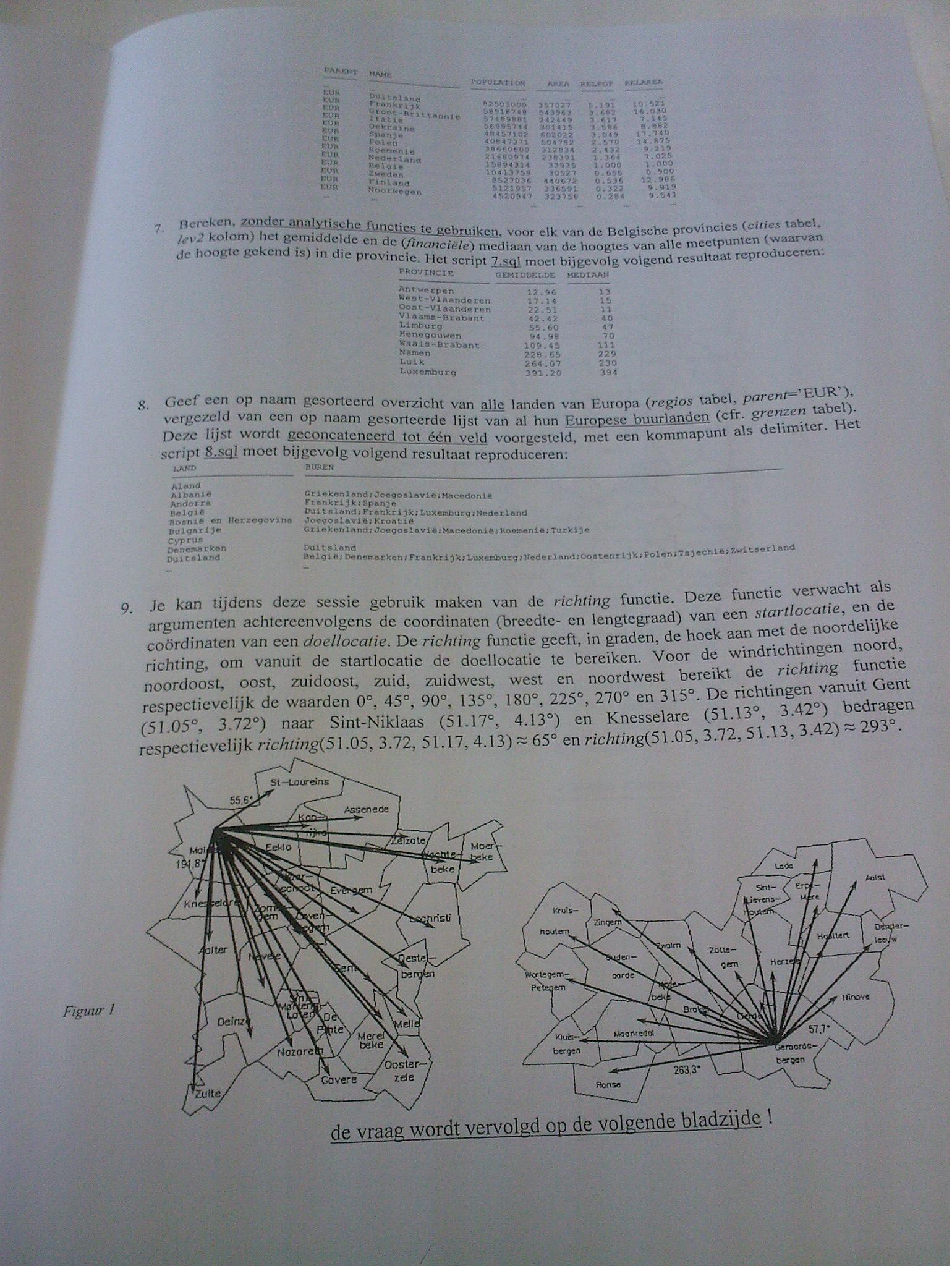
* transactie locking, wa uitleg geven + hoe? + welke niveaus + wa ist probleem dermee
* welke soorten triggers bestaan er? + wanneer gebruikte wa
* Waarom worden Procedurele extenties (3GL) minder gebruikt dan vroeger? waarom? wanneer mag men ze wel nog gebruiken? + uitleg bij deze gevallen

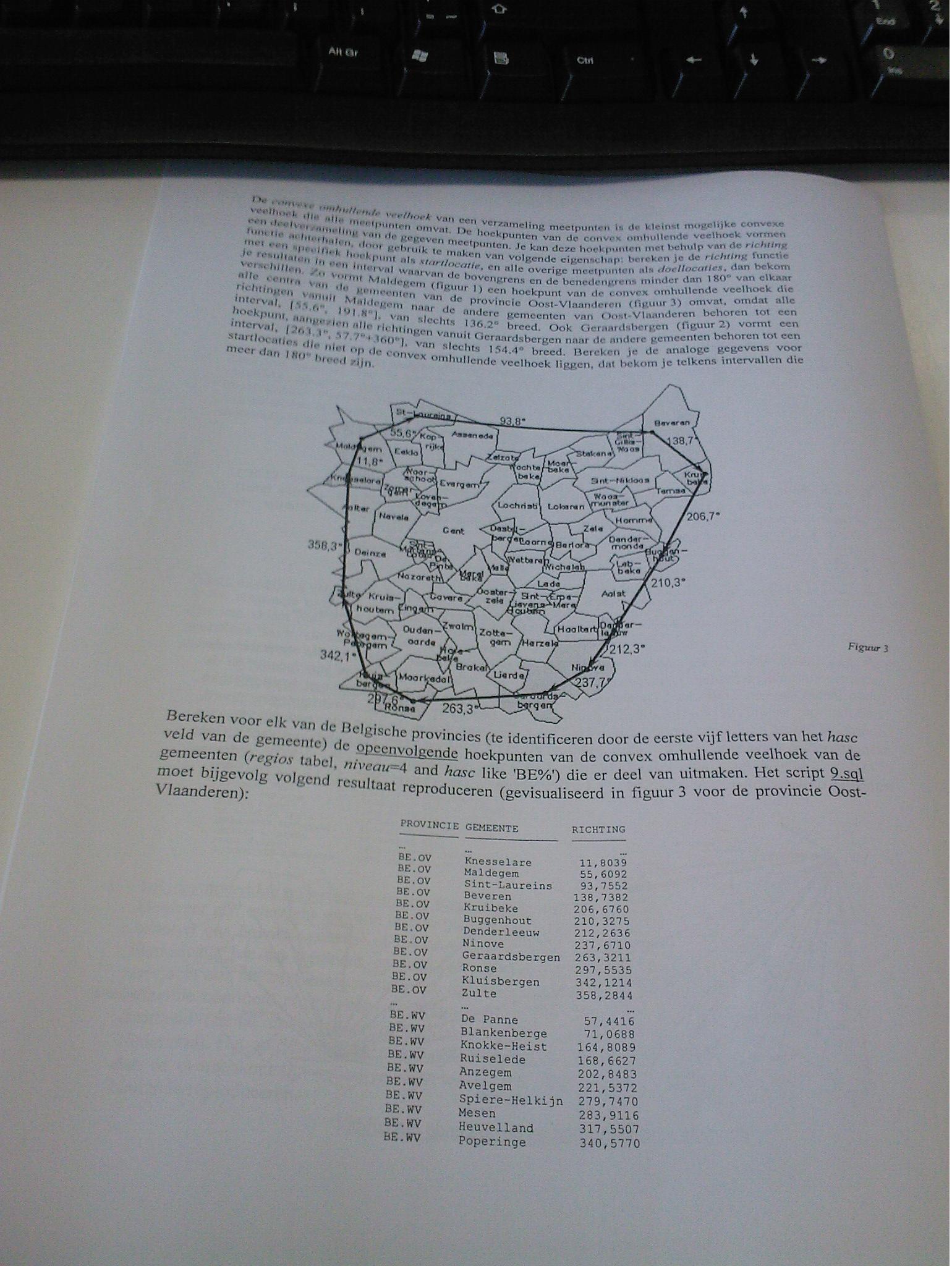
Bijvragen:

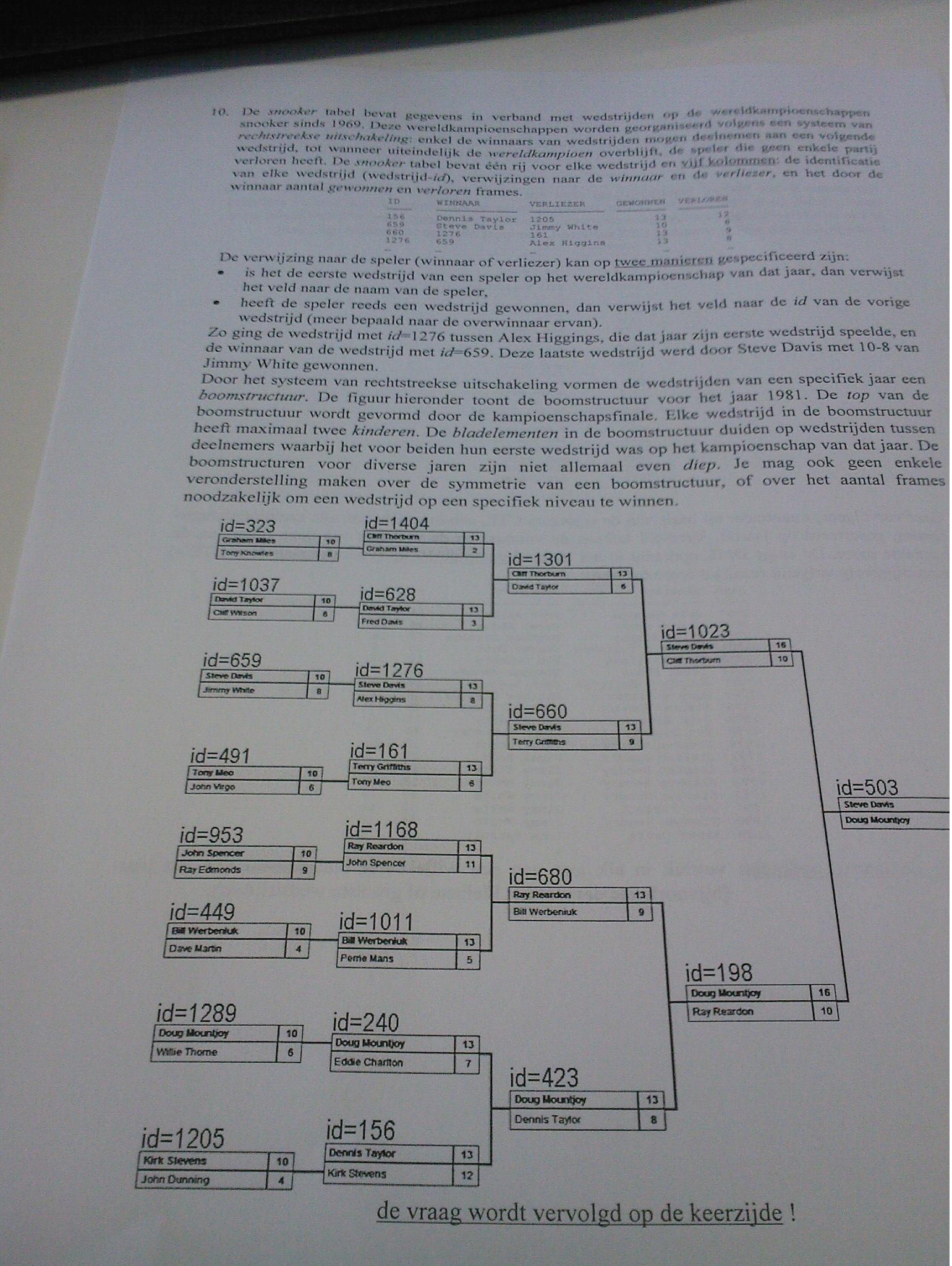
* Wanneer zijn gedenormaliseerde tabellen wel verantwoord?
* Wat is granulariteit?

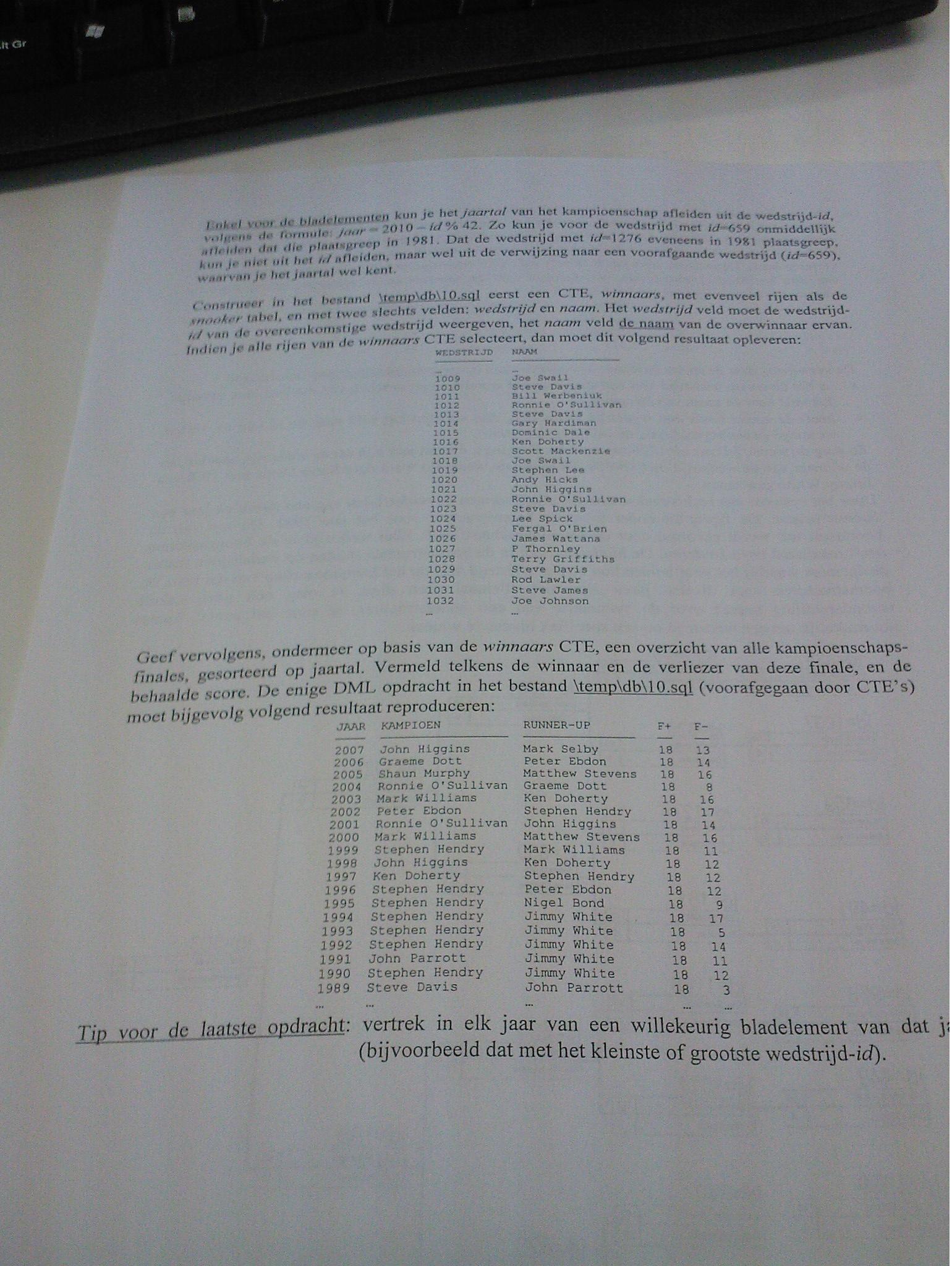
Oefeningen

****









**Oplossing oefening 3:**

Kan de opgave niet echt volledig lezen: dit geeft dezelfde output als zijn oplossing is maar ik weet niet op welke manier hij bepaalt wanneer een naam in een bep kolom mag ik denk >= 10 per discipline en >=25 voor totaal

with y as (

select name, discipline,count(1) aantal,

row\_number() over(partition by discipline order by count(1) DESC ) nummer

from results join races on results.rid = races.rid

where rank = 1 group by name, discipline having count(1) >= 10

), som as (

select name, count(1) totaal,

row\_number() over( order by count(1) DESC) plaats

from results join races on results.rid = races.rid

where rank = 1

group by name

having count(1) >= 25

), x as (

select name,totaal,row\_number() over(order by name) bla,

max(plaats) over() max from som )

select

(select name from y where discipline = 'DH' and nummer = zz.bla) DH,

(select aantal from y where discipline = 'DH' and nummer = zz.bla) pts,

(select name from y where discipline = 'SG' and nummer = zz.bla) SG,

(select aantal from y where discipline = 'SG' and nummer = zz.bla) pts,

(select name from y where discipline = 'GS' and nummer = zz.bla) GS,

(select aantal from y where discipline = 'GS' and nummer = zz.bla) pts,

(select name from y where discipline = 'SL' and nummer = zz.bla) SL,

(select aantal from y where discipline = 'SL' and nummer = zz.bla) pts,

(select name from som where plaats = zz.bla) totaal,

(select totaal from som where plaats = zz.bla) Ml

from x zz where zz.bla <= zz.max

**Oplossing oefening 4:**

Deze vond ik zeer moeilijk, na lang werken klopt hij, maar er moeten gemakkelijkere manieren bestaan!

with x as (select name, racedate,gender,discipline from races ra join results re on ra.rid = re.rid where rank = 1

)

, a as (select x1.name, x1.racedate date1,x1.gender, x1.discipline, max(x2.racedate) date2

from x x1

join x x2 on x1.gender = x2.gender

and x1.discipline = x2.discipline

and x1.name = x2.name

and x1.racedate<=x2.racedate

left join x x3 on x1.gender =x3.gender

and x1.discipline = x3.discipline

and x2.name = x3.name

and x3.racedate between x1.racedate+1 and x2.racedate-1

where x3.racedate is null

group by x1.name,x1.racedate,x1.gender,x1.discipline

order by x1.racedate )

, y as (

select name,date1,gender,discipline,date2, row\_number() over(partition by discipline,gender order by date1) nummer from a

)

, z as (

select y1.name, y1.date1,y1.date2, gender,discipline,

case when ( select name from y where y1.discipline = discipline and gender = y1.gender and nummer = y1.nummer-1) = y1.name then 1 else 0 end nieuwe

from y y1 )

, q as (

select z1.name, z1.date1, gender, discipline,

( select 1+ count(case nieuwe when 0 then 1 end) from z where date1<= z1.date1 and discipline = z1.discipline and gender = z1.gender ) groepen

from z z1

)

select name, min(date1), max(date1),count(1),gender,discipline

from q

group by name, gender,discipline,groepen

having count(1) >=5

order by 4 DESC,3 DESC

**Oplossing oefening 5:**

with x as (select re.name naam,

ra.rid,ra.racedate from results re join races ra on re.rid = ra.rid where re.rank = 1

) , y as (

select x1.naam naam, x1.racedate min,x1.rid minrid , x2.racedate max,x2.rid maxrid,

x2.racedate-x1.racedate verschil, mod((x2.racedate-x1.racedate),12) jaren

from x x1

join x x2 on x1.naam = x2.naam and x2.racedate> x1.racedate

left join x x3 on x1.naam = x3.naam and x3.racedate between x1.racedate +1 and X2.racedate -1

where x3.racedate is null

) ,z as (

select y1.verschil,y1.naam,y1.min,y1.minrid,y1.max,y1.maxrid, extract(year from y1.max)- extract(year from y1.min) ||'j '|| mod(y1.verschil,365) ||' d' delta

from y y1

, y y2

group by y1.naam,y1.min,y1.max,y1.verschil,y1.minrid, y1.maxrid

having sum (case when y1.verschil <= y2.verschil then 1 end ) <=20

)

select naam, min, (select resort from races where rid = minrid) as " ", max, (select resort from races where rid = maxrid) as " ", delta

from z order by verschil DESC

**Oplossing oefening 6:**

with landen as (

select parent,name,population,area

,row\_number() over(partition by parent order by population DESC) nrpop

,row\_number() over(partition by parent order by area DESC) nrarea

from regios where niveau = 0 and population is not null and area is not null

), top as (

select parent, name,population,area

,case population/area when max( population/area) over(partition by parent) then population end relpop

,case population/area when max( population/area) over(partition by parent) then area end relarea

from landen where nrpop<=10 OR nrarea <=10

)

select parent,name,population,area,

cast( population/max(relpop) over(partition by parent) as numeric(9,3)) relpop,

cast( area/max(relarea) over(partition by parent) as numeric(9,3)) relarea

from top

**Oplossing oefening 7:**

with x as (select lev2 provincie, cast ( avg(elevation) as numeric( 9,2) ) gemiddelde

from cities

where iso = 'BE' and lev2 is not null and elevation is not null

group by lev2

)

select x.lev2 provincie , ( select gemiddelde from x where provincie = x.lev2 ) gemiddelde ,x.elevation mediaan

from cities x join cities y on x.iso = y.iso and x.lev2 = y.lev2

where x.iso = 'BE' and x.lev2 is not null and x.elevation is not null and y.elevation is not null

group by x.lev2,x.elevation

having abs(count( case when x.elevation < y.elevation then 1 end )

- count( case when x.elevation > y.elevation then 1 end ) )

<= count(case when x.elevation = y.elevation then 1 end)

order by 2

**Oplossing oefening 8:**

with x as (

select r1.name land ,r2.name buur,row\_number() over (partition by r1.name order by r2.name) nummer, count(1) over(partition by r1.name) totaal

from regios r1, regios r2

where (r1.parent = 'EUR'

and r2.parent = 'EUR')

and ( (r1.hasc,r2.hasc) in (select hasc1, hasc2 from grenzen)

or (r2.hasc,r1.hasc) in (select hasc1, hasc2 from grenzen) )

)

, y as (

select totaal,level levl,land, ltrim(sys\_connect\_by\_path(buur,';'),';') buren

from x

start with nummer = 1 connect by prior nummer +1 = nummer and prior land = land )

select land, buren from y where levl = totaal

Oplossing is NIET 100% gelijk aan zijn gegeven oplossing, de reden hiervoor ligt volgens mij aan de aanpassingen van GEO tabel

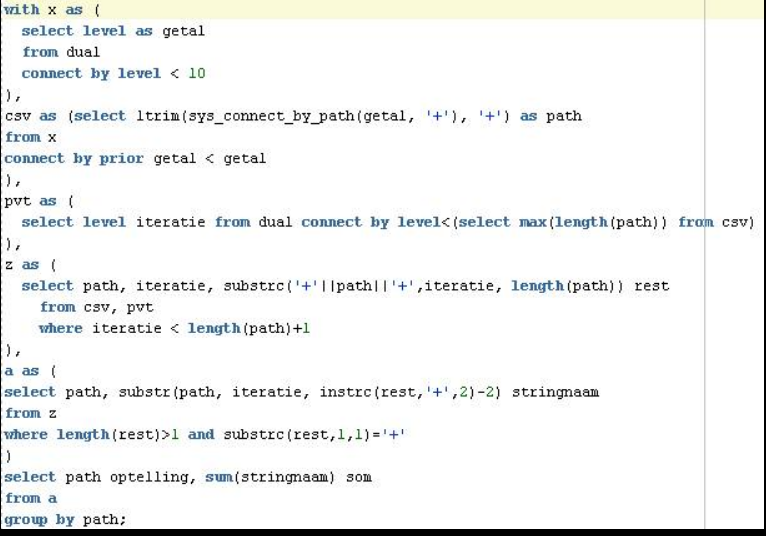
vb: Bulgarije heeft in zijn voorbeeld Turkije als buur, maar als we in volgend stukje code gaan kijken: ... ligt turkije in AZIE

select name, parent from regios where name = 'Turkije'

Zelf dingen aangevuld van fb groep

Heeft er iemand een idee hoe de examenvraag van de partities (recursieve vraag van het examen) op te lossen?

Start met een CTE die de getallen 1 tot en met 9 maakt.   
dan die cte recursief aanspreken



Juist de sortering is nog niet gemaakt. Maar dat zou normaal niet zo veel werk mogen zijn.

/////////////////////

Oefening kalender:

1. with days as(
2. select
3. last\_day(sysdate-extract(day from sysdate))+level x
4. from dual
5. connect by last\_day(sysdate-extract(day from sysdate)) + level <=last\_day(sysdate-extract(day from sysdate)+1)
6. )
7. ,cal as(
8. select x,
9. extract(year from x) as jaar,
10. extract(month from x) as maand,
11. to\_char(x,'iw') as weeknr, -- code iw: week of the year
12. to\_char(x,'dd') as dagmaand, -- code dd: day of the month
13. to\_char(x,'d') as dagnr -- code d: day of the week
14. from days
15. )
17. select substr('janfebmaraprmeijunjulaugsepoktnovdec',3\*min(maand)-2,3) "Maand",
18. max(case dagnr when '1' then dagmaand else ' ' end) ma,
19. max(case dagnr when '2' then dagmaand else ' ' end) di,
20. max(case dagnr when '3' then dagmaand else ' ' end) wo,
21. max(case dagnr when '4' then dagmaand else ' ' end) do,
22. max(case dagnr when '5' then dagmaand else ' ' end) vr,
23. max(case dagnr when '6' then dagmaand else ' ' end) za,
24. max(case dagnr when '7' then dagmaand else ' ' end) Zo
26. from cal
27. group by jaar,maand,weeknr
28. order by jaar,maand,max(dagmaand)

//////////////////////////

Oef2 van 2014/2015

1. **WITH** topN **AS** (
2. **SELECT** ra.discipline, re.name, ra.gender, **COUNT**(1) **AS** overwinningen, **ROW\_NUMBER**() **OVER**(partition **BY** ra.discipline **ORDER** **BY** **COUNT**(1) **DESC**)**AS** rank
3. , **ROW\_NUMBER**() **OVER**(partition **BY** ra.gender, ra.discipline **ORDER** **BY** **COUNT**(1) **DESC**) **AS** nr
4. **FROM** races ra **JOIN** results re **ON** ra.rid=re.rid
5. **WHERE** re.rank=1
6. **GROUP** **BY** ra.discipline, re.name, ra.gender
7. **ORDER** **BY** discipline, overwinningen **DESC**
8. ), x **AS** (
9. **SELECT** topN.\*, **CASE** **WHEN** nr <=3 **THEN** 3
10. **WHEN** nr **BETWEEN** 3 **AND** 7 **THEN** 7
11. **WHEN** nr **BETWEEN** 7 **AND** 10 **THEN** 10 **END** **AS** top
12. **FROM** topN
13. ), y **AS** (
14. **SELECT** x.\*, **ROW\_NUMBER**() **OVER**(partition **BY** discipline, top **ORDER** **BY** overwinningen **DESC**) **AS** format
15. **FROM** x
16. **WHERE** top **IS** **NOT** **NULL**
17. **ORDER** **BY** discipline, top, rank
18. )
19. **SELECT** **CASE** **WHEN** format=1 **THEN** discipline **ELSE** ' ' **END** **AS** discipline\_
20. , **MAX**(**CASE** **WHEN** top=3 **THEN** name **ELSE** ' ' **END**) **AS** top3, **MAX**(**CASE** **WHEN** top=3 **THEN** to\_char(overwinningen) **ELSE** ' ' **END**) **AS** pt\_1
21. , **MAX**(**CASE** **WHEN** top=7 **THEN** name **ELSE** ' ' **END**) **AS** top7, **MAX**(**CASE** **WHEN** top=7 **THEN** to\_char(overwinningen) **ELSE** ' ' **END**) **AS** pt\_2
22. , **MAX**(**CASE** **WHEN** top=10 **THEN** name **ELSE** ' ' **END**) **AS** top10, **MAX**(**CASE** **WHEN** top=10 **THEN** to\_char(overwinningen) **ELSE** ' ' **END**) pt\_3
23. **FROM** y
24. **GROUP** **BY** discipline, format
25. **ORDER** **BY** discipline, format

////////////////

Oef me door afrika reizen:

with belangrijk as(  
select hasc1, hasc2  
from grenzen  
UNION  
select hasc2, hasc1  
from grenzen  
), landen as(  
SELECT name land, hasc hasc1, hasc2  
FROM regios  
join belangrijk  
on hasc=hasc1  
WHERE parent = 'SAM'  
), trips as(  
SELECT land, hasc1, hasc2, level niveau, ltrim(sys\_connect\_by\_path(land,';'),';') pad, max(level) over() max  
FROM landen  
WHERE hasc2 = 'BR'  
START WITH hasc1 = 'BR'  
CONNECT BY nocycle prior hasc2 = hasc1  
)  
SELECT pad  
FROM trips  
WHERE max=niveau;

/////////////////////

13/14 oef5

1. with comp as(
2. select name,
3. season,
4. discipline,
5. sum(points) as punten,count(1) over(partition by name,discipline) as totaal,
6. row\_number() over(partition by name,discipline order by season) as aantal
8. from ranking
9. where season between 2006 and 2010
10. and discipline in ('DH','SG','GS','SL','KB')
11. group by name,discipline,season
12. order by name,discipline,season
13. )
15. ,tss as(
16. select name,
17. season,
18. discipline,
19. ltrim(sys\_connect\_by\_path(punten,'+'),'+') as totaal
20. from comp
21. where level=totaal
22. start with aantal=1
23. connect by prior aantal=aantal-1 and prior name=name and prior discipline=discipline
24. )
26. select name,season,
27. max(case when discipline='DH' then totaal else ' ' end) "DH",
28. max(case when discipline='SG' then totaal else ' ' end) "SG",
29. max(case when discipline='GS' then totaal else ' ' end) "GS",
30. max(case when discipline='SL' then totaal else ' ' end) "SL",
31. max(case when discipline='KB' then totaal else ' ' end) "KB"
32. from tss
33. group by name,season
34. order by name,season

//////////////

14/15 oef 7

1. with x as(
2. select 6\*level-1 x
3. from dual
4. connect by level-1<300)
6. ,y as(select 3\*level-1 x
7. from dual connect by level-1<600)
9. ,z as (select x from x union select x from y)
11. ,rec as(
12. select level x,ntile(60) over(order by level) as helerij
13. from dual
14. connect by level-1 < 1800)
16. ,mix as(
17. select z.x,helerij
18. from z join rec on z.x <= helerij\*30  and z.x=rec.x)
20. -- hierboven is in generation van een sequentie omdat ik zijn opgegeven CTE niet heb
22. ,res as(
23. select 'rij '||u.helerij as rijnr, u.x as sta ,v.x as einde,v.x-u.x-1 as vrij
24. from mix u join mix v on u.helerij=v.helerij and v.x > u.x
25. where not exists (select null from mix where x between u.x+1 and v.x-1)
26. order by u.x,v.x
27. )
28. -- vrij is altijd 2 maar dat is omdat mijn gegeneerde plaatsen een rekenkundige rij zijn
30. select to\_char(rijnr) rij ,to\_char(sta) "[..",to\_char(einde) "..]",vrij
31. from res
32. union
33. select 'totaal','--','--',sum(vrij) over()
34. from res

/////////////////////////////

14/15 oef 5

1. select hasc,name,
2. (select count(1) from regios
3. where parent=p.hasc) "#k",
5. (select count(1)
6. from regios q
7. where (select parent from regios where q.parent=hasc)=p.hasc) "#kk",
9. (select count(1)
10. from regios q
11. where (select parent from regios r where (select parent from regios where r.parent=hasc)=hasc)=p.hasc) "#kkk"
13. from regios p
14. where hasc in(select hasc from members where afkorting='EU')
15. ;
17. -- zeer vreemd dat het met de cities tabel zou moeten, die heeft geen data voor lev3 toch niet voor belgië (achterkleinkind)
18. select iso
19. ,(select count(distinct lev1) from cities where c.iso=iso) "#k"
20. ,(select count(distinct lev2) from cities where c.iso=iso) "#kk"
21. ,(select count(distinct lev3) from cities where c.iso=iso) "#kkk"
22. from cities c
23. where iso='BE'
24. group by iso

//////////////////

14/15 oef 3

1. with x as(
2. SELECT  r1.name,x.resort,x.racedate as vorige ,min(y.racedate) as volgende
4. FROM Races x
5. JOIN Results r1 ON x.rid=r1.rid
6. AND r1.rank=1
7. JOIN Races y ON y.racedate>x.racedate
8. JOIN Results r2 ON y.rid=r2.rid
9. AND r1.name=r2.name
10. AND r2.rank=1
11. WHERE x.discipline in ('DH','SG','GS','SL','KB')  AND y.discipline in ('DH','SG','GS','SL','KB')
12. GROUP BY r1.name,x.resort,x.racedate
13. ORDER BY r1.name,x.resort,x.racedate)
15. ,y as(
16. select x.vorige,r.racedate,x.volgende,re.name,r.gender,r.resort,r.discipline,re.rank
17. , **count(distinct r.rid**) over(partition by x.vorige,x.volgende,re.name,r.gender) as podium
18. ,row\_number() over(partition by x.vorige,re.cid order by r.racedate ) as format
19. from x
20. join races r on r.racedate between x.vorige and x.volgende and x.resort <> r.resort
21. join results re on re.rid=r.rid and x.name=re.name
22. where re.rank in (2,3)
23. order by podium desc,format
24. )

27. select
28. case when y.format=1 then y.name else ' ' end "naam",
29. y.vorige "vorige winst",
30. y.racedate "racedate",
31. y.resort "resort",
32. y.discipline "di",
33. y.rank "rank",
34. y.volgende "volgende winst"
36. from y
37. left join y z on y.podium < z.podium and y.gender=z.gender
38. where z.podium is null