**FAQ reeks 2: combinatie van meerdere tabellen**

**Algemene opmerkingen**

* Typisch krijgen we veel vragen met betrekking tot welk type join gebruikt moet/kan worden. Als hulp zullen we hieronder een aantal tips oplijsten die je in staat zouden moeten stellen om het juiste type join te bepalen voor elke oefening.
  + In *bijna alle gevallen* moet je een INNER JOIN gebruiken, zéker wanneer je verschillende tabellen wil koppelen op basis van de vreemde sleutel-referenties tussen deze tabellen. Vergeet niet om je tabellen in dit geval steeds te joinen op de *volledige* vreemde sleutel (bv. indien de vreemde sleutel uit meerdere attributen bestaat) als je correcte resultaten wil bekomen (denk eens na waarom dit zo belangrijk is en wat er gebeurt als je dit niet zou doen).
  + In gevallen waarbij je wil weten of iets bestaat zonder iets anders (bv. radioprogramma’s zonder presentator, liedjes die nooit zijn afgespeeld,...) kan je steeds een LEFT/RIGHT JOIN gebruiken met een ‘IS NULL’-check in de WHERE-clausule.
  + Zowel CROSS JOIN als FULL JOIN worden amper gebruikt in de praktijk aangezien dit zeer zware operaties zijn en deze operaties in de meeste gevallen vervangen kunnen worden door een ander type join. Enkel indien je alle combinaties van rijen uit twee of meerdere tabellen wil verkrijgen kunnen deze types joins van pas komen.
* Als je meerdere tabellen gaat combineren kan je best een alias geven aan elke tabel in de FROM-clausule van de query (bv. ... FROM radioprogram\_presenter rp INNER JOIN presenter p ...) in plaats van steeds opnieuw de volledige tabelnaam te schrijven. Bovendien is het gebruik van aliasen noodzakelijk als je dezelfde tabel meerdere keren gebruikt in eenzelfde FROM-clausule (bv. ... FROM presenter p1 INNER JOIN presenter p2 ...).

**Radio 2.4**

* Veel studenten maken fouten omdat ze niet weten hoe patronen in date/time functies gebruikt kunnen worden. Stel, bijvoorbeeld, dat je het attribuutstarttime van de tabel radiobroadcast wil weergeven in het formaat ‘17/10/2011’ (je wil dus enkel de datum, zonder tijdsaanduiding, in een specifiek formaat behouden), dan kan dit met volgende query: SELECT to\_char(rb.starttime, ‘DD/MM/YYYY’) FROM radiobroadcast rb; Alle details met betrekking tot date/time patronen kunnen teruggevonden worden in de PostgreSQL documentatie. Links naar de relevante webpagina’s zijn dan weer terug te vinden in de introductiepresentatie van reeks 1.
* Kan je in de PostgreSQL documentatie terugvinden wat het patroon is voor ‘minuten’ en wat het patroon is voor ‘maanden’. Zijn deze patronen gelijk?
* Sorteer je de resultaten op begintimestamp of op starttime?

**Radio 2.6**

* Dit is een typische oefening waarvoor je gebruik kan maken van een LEFT/RIGHT JOIN in combinatie met een ‘IS NULL’-check in de WHERE-clausule.

**Radio 2.7**

* Maak je gebruik van het correcte type join? Je moet elk nummer van Daan behouden, ook al komt het niet voor als playlistitem.
* De nummers van ‘Daan’ werden niet telkens om 23u00 afgespeeld. Gebruik je de starttijd van de uitzending waarin het nummer werd afgespeeld of de exacte tijd waarop het nummer werd afgespeeld tijdens een uitzending?
* Sorteer je de resultaten op een correcte manier? Aan een ORDER BY-clausule kunnen er een aantal (optionele) parameters/opties worden meegegeven. Voorbeelden hiervan staan opgelijst in de introductiepresentatie van reeks 1 en in de PostgreSQL documentatie.
* Selecteer je de titel van het nummer of de titel van het radioprogramma waarin het nummer werd afgespeeld?

**Radio 2.8**

* Om koppels van presentatoren te bekomen moet je de tabel presenter met zichzelf joinen.
* Bij deze oefening is de join-conditie ietwat speciaal, aangezien de oefening vraagt om koppels van presentatoren te maken *van wie de voornaam begint met hetzelfde karakter*.
* Om, voor elke rij, de waarde in kolom fullname1 alfabetisch voor de waarde in kolom fullname2 te ‘sorteren’, kan je geen gebruik maken van de ORDER BY-clausule, aangezien dit de rijen sorteert, eerst op basis van de waarden in kolom fullname1 en dan (in het geval van gelijke waarden in de kolom fullname1) op basis van de waarden in kolom fullname2. Wel kan je gebruik maken van p1.firstname || ‘ ‘ || p1.lastname < p2.firstname || ‘ ‘ || p2.lastname in de WHERE-clausule (of in de JOIN-conditie). Deze booleaanse expressie houdt namelijk enkel rijen over waarin de volledige naam van presentator 1 alfabetisch voor de volledige naam van presentator 2 ligt. Meer nog, het zal ook rijen verwijderen waarin de volledige naam van presentator 1 gelijk is aan de volledige naam van presentator 2.

**Radio 2.9**

* Deze vraag kan je herformuleren als volgt: Geef alle radioprogramma’s (1) waarvoor er geen tweede presentator bestaat en (2) die gepresenteerd zijn door ‘Ayco Duyster’.
* Om deze oefening correct op te lossen is het opnieuw noodzakelijk om de tabel radioprogram\_presenter met zichzelf te joinen.
  + In eerste instantie moet je koppels maken van rijen uit de tabel radioprogram\_presenter waarvoor het eerste radioprogramma gelijk is aan het tweede radioprogramma (op basis van een gelijk radioprogramid), maar de presentator van het eerste radioprogramma verschilt van de presentator van het tweede radioprogramma (op basis van een verschillend presenterid). Dit geeft dus alle koppels van gelijke radioprogramma’s met verschillende presentatoren.
  + Daarna dien je enkel de koppels (rijen) te behouden waarvoor er geen tweede presentator bestaat (WHERE p2.presenterid IS NULL).
  + Tot slot dien je in deze tabel enkel de radioprogramma’s over te houden die gepresenteerd worden door ‘Ayco Duyster’. Let op, het is mogelijk dat er meer dan een presentator is met de naam ‘Ayco Duyster’...

**Radio 2.10**

* Je moet een attribuut van het datatype *date* (birthdate) vergelijken met een attribuut van het datatype *timestamp* (starttime). Dit kan je doen door ...
  + ... beide attributen *string*-gewijs te vergelijken, na omzetting van de waarden van beide attributen naar het *string* datatype.
  + … de dag en de maand van beide attributen te extraheren en ze apart te vergelijken met elkaar.
* Let op, de functies *extract* en *date\_part* kunnen slechts één veld tegelijk uit een waarde van datatype timestamp extraheren.
* Aangezien je moet vergelijken met een zekere verjaar*dag* (dus dag van de maand en maand) dien je het jaartal niet in rekening te brengen bij het vergelijken.

**Garage 2.6**

Zoals je wel al weet kan je eender welke tabellen met elkaar joinen op basis van eender welke booleaanse conditie (dus niet enkel op basis van vreemde sleutel-referenties). Dit kan soms wel tot vreemde resultaten leiden. In de praktijk worden daarom de meeste joins uitgevoerd op basis van gelijkheid van waarden voor vreemde sleutel-attributen. Let op, bij deze oefening dien je echter wel te joinen op basis van een ander soort conditie (de waarden voor postcode *of* gemeente moeten gelijk zijn).