










































# Beheer De Oefeningen













Voeg een [nieuw record](#) toe.

| academiejaar | oefeningreeks | opgave  | oplossing   |   |
|--------------|---------------|---|---|---|
|              |               | Maak een lijst met alle spelers die in Zoetermeer wonen.  | SELECT spelersnr, naam, jaartoe<br>FROM spelers<br>WHERE plaats = 'Zoetermeer';   |    |
|              |               | Maak een lijst met alle teams waarvan speler 27 de aanvoerder is.   | SELECT *<br>FROM teams<br>WHERE spelersnr = 27;   |    |
|              |               | Maak een lijst met alle mannelijke spelers.   | SELECT spelersnr, naam, jaartoe<br>FROM spelers<br>WHERE geslacht = 'M';  |    |
|              |               | Maak een lijst van alle wedstrijden, die gespeeld werden voor het tweede team (teamnr: 2) en die gewonnen zijn door een speler van onze club (let op de benamingen van de kolomkoppen).   | SELECT wedstrijdnr, spelersnr, gewonnen AS<br>gewonnen_sets, verloren AS verloren_sets<br>FROM wedstrijden<br>WHERE teamnr = 2 AND gewonnen > verloren;   |    |
|              |               | Maak een overzicht van alle wedstrijden van speler met nummer 6 waarbij je berekent met welk aantal sets verschil hij de wedstrijd gewonnen of verloren heeft.  | SELECT spelersnr, gewonnen, verloren, gewonnen -<br>verloren AS verschil<br>FROM wedstrijden<br>WHERE spelersnr = 6;  |    |
|              |               | Maak een lijst van alle actieve bestuursleden.<br>Een actief bestuurslid is een bestuurslid met een lege eind_datum.<br>Sorteer op begin_datum oplopend.  | SELECT spelersnr, functie<br>FROM bestuursleden<br>WHERE eind_datum IS NULL<br>ORDER BY begin_datum;  |    |
|              |               | Maak een lijst met de wedstrijden van speler nummer 6 waarbij je aangeeft of de wedstrijd gewonnen (toon: 'gewonnen') of verloren (toon: verloren) werd door de betreffende speler.   | SELECT spelersnr, wedstrijdnr, gewonnen, verloren,<br>CASE<br>WHEN gewonnen > verloren THEN 'gewonnen'<br>ELSE 'verloren'<br>END AS resultaat<br>FROM wedstrijden<br>WHERE spelersnr = 6;   |  |
|              |               | Maak een lijst van alle boetes met een boetebedrag boven 75 euro die betaald werden in 1980, 1981, 1982 of 1983.<br><br>Let op:<br>1) de titels van de kolommen moeten gebruikersvriendelijk zijn en geen afkortingen bevatten (spelersnr wordt speler, jaar van de boete verschijnt als jaar en het bedrag wordt weergegeven als bedrag).<br>2) Geef geen date als jaar terug, maar een int. | SELECT spelersnr AS speler, EXTRACT(YEAR FROM<br>datum) AS jaar, bedrag<br>FROM boetes<br>WHERE bedrag > 75<br>AND EXTRACT(YEAR FROM datum) BETWEEN 1980<br>AND 1983  |  |
|              |               | Geef het totaal aantal boetes met een bedrag groter dan 50.   | SELECT COUNT(*) AS aantal_boetes<br>FROM boetes<br>WHERE bedrag > 50;   |  |
|              |               | Maak een overzicht waarbij je per woonplaats aangeeft hoeveel spelers er wonen in die gemeente.<br>Sorteer op plaats.   | SELECT plaats, COUNT(*) AS aantal<br>FROM spelers<br>GROUP BY plaats<br>ORDER BY plaats;  |  |
|              |               | Ga na hoeveel mannelijke en hoeveel vrouwelijke spelers de club telt. Zorg ervoor dat je lijst gesorteerd wordt van het grootst aantal spelers naar het kleinste aantal.  | SELECT geslacht, COUNT(*) AS aantal<br>FROM spelers<br>GROUP BY geslacht<br>ORDER BY aantal DESC;   |  |
|              |               | Maak een lijst van alle vrouwelijke aanvoerders van een team. Hierbij toon je voor deze spelers het spelersnummer en de volledige naam.   | SELECT spelers.spelersnr, spelers.naam,<br>spelers.voorletters<br>FROM spelers, teams<br>WHERE spelers.spelersnr = teams.spelersnr<br>AND spelers.geslacht = 'V';   |  |
|              |               | Geef een lijst met het spelersnummer, de naam van de speler, de datum van de boete en het bedrag van de boete van al de spelers die een boete gekregen hebben met een bedrag groter dan 45,50 euro en in Rijswijk wonen.<br>Sorteer op spelersnr en het volgnummer van de boete.<br>Gebruik expliciete joins.   | SELECT spelers.spelersnr, spelers.naam, boetes.datum,<br>boetes.bedrag<br>FROM spelers<br>INNER JOIN boetes ON spelers.spelersnr =<br>boetes.spelersnr<br>WHERE boetes.bedrag > 45.50<br>AND spelers.plaats = 'Rijswijk'<br>ORDER BY spelers.spelersnr, boetes.betalingsnr;   |  |
|              |               | Maak een lijst van alle mannelijke aanvoerders van een team. Geef voor die aanvoerders de wedstrijden waarin ze gespeeld hebben. Geef ook aanvoerders die geen wedstrijd gespeeld hebben. Hierbij toon je voor deze spelers het spelersnummer en de volledige naam, voor het team de divisie en voor de wedstrijd het wedstrijdnummer.<br>Sorteer aflopend op het wedstrijdnummer.            | SELECT spelers.spelersnr, spelers.naam,<br>spelers.voorletters, teams.divisie, wedstrijden.wedstrijdnr<br>FROM spelers<br>INNER JOIN teams ON spelers.spelersnr =<br>teams.spelersnr<br>LEFT OUTER JOIN wedstrijden ON spelers.spelersnr =<br>wedstrijden.spelersnr<br>WHERE spelers.geslacht = 'M'<br>ORDER BY wedstrijden.wedstrijdnr DESC; |  |
|              |               |   | SELECT datum, bedrag  |   |








|  |  |  |   |   |
|--|--|--|---|---|
|  |  | Geef een lijst met alle boetes kleiner dan 75 euro.<br>Zorg dat de oudste boete bovenaan staat.  | FROM boetes<br>WHERE bedrag < 75<br>ORDER BY datum;   |     |
|  |  | Maak een lijst met alle spelers die niet in Zoetermeer wonen en die na 1980 zijn toegetreden tot de club.<br>Zorg dat de jongste speler bovenaan staat   | SELECT spelersnr, naam, voorletters<br>FROM spelers<br>WHERE plaats <> 'Zoetermeer'<br>AND jaartoe > 1980<br>ORDER BY geb_datum DESC;   |    |
|  |  | Geef een lijst van wedstrijden die gespeeld zijn door team 1 en waarbij ofwel het aantal gewonnen of het aantal verloren sets gelijk is aan nul.   | SELECT wedstrijdnr, teamnr<br>FROM wedstrijden<br>WHERE (gewonnen = 0 OR verloren = 0)<br>AND teamnr = 1;   |    |
|  |  | Geef een lijst met het spelersnummer, de naam van de speler, de datum van de boete van al de spelers die in 1980 een boete gekregen hebben en in Rijswijk wonen.<br><br>Zorg voor de juiste opmaak van de datum in het resultaat (niet YYYY-MM-DD maar DD/MM/YYYY) waarbij je de to_char functie gebruikt.<br><br>Sorteer op spelersnr.  | SELECT spelers.spelersnr, voorletters, naam,<br>to_char(boetes.datum, 'DD/MM/YYYY') AS boetedatum<br>FROM spelers<br>INNER JOIN boetes ON spelers.spelersnr =<br>boetes.spelersnr<br>WHERE spelers.plaats = 'Rijswijk'<br>AND EXTRACT(YEAR FROM boetes.datum) = 1980<br>ORDER BY spelers.spelersnr;   |    |
|  |  | Maak een lijst van alle spelers die al een wedstrijd gespeeld hebben.<br>Alleen spelers met een 'e' (hoofdletter of kleine letter) in de naam moeten getoond worden. Zorg dat deze laatste voorwaarde met één WHERE-conditie gerealiseerd wordt.<br>Let ook op de juiste kolomhoofdingen.<br>Sorteer op spelersnr en zorg dat elke speler maar één keer voorkomt.<br>Gebruik geen concat, maar wel   . | SELECT DISTINCT spelers.spelersnr, spelers.voorletters<br>   ' '    spelers.naam AS spelersnaam<br>FROM spelers, wedstrijden<br>WHERE lower(spelers.naam) like '%e%' AND<br>spelers.spelersnr = wedstrijden.spelersnr<br>ORDER BY spelers.spelersnr;  |    |
|  |  | Geef alle boetes groter dan 25 euro voor aanvoerders die meegespeeld hebben in een winnende wedstrijd met het team waarvan ze aanvoerder zijn.<br>Sorteer op datum van de boete (de tekstuele waarde, niet de echte datum).  | SELECT DISTINCT wedstrijden.spelersnr,<br>boetes.betalingsnr, TO_CHAR(boetes.datum,<br>'DD/MM/YYYY') AS boetedatum<br>FROM teams, wedstrijden, boetes<br>WHERE wedstrijden.spelersnr = teams.spelersnr<br>AND teams.teamnr = wedstrijden.teamnr<br>AND wedstrijden.spelersnr = boetes.spelersnr<br>AND boetes.bedrag > 25<br>AND wedstrijden.gewonnen > wedstrijden.verloren<br>ORDER BY boetedatum;  |    |
|  |  | Geef een overzicht van alle divisies waar wedstrijden voor gespeeld zijn met spelers die actief bestuurslid zijn en minstens één boete hebben.<br>Zorg dat elke divisie maar één keer voorkomt en sorteer de lijst alfabetisch.  | SELECT DISTINCT teams.divisie<br>FROM teams<br>INNER JOIN wedstrijden ON teams.teamnr =<br>wedstrijden.teamnr<br>INNER JOIN bestuursleden ON bestuursleden.spelersnr =<br>wedstrijden.spelersnr<br>INNER JOIN boetes ON wedstrijden.spelersnr =<br>boetes.spelersnr<br>WHERE bestuursleden.eind_datum IS NULL<br>ORDER BY teams.divisie;  |  |
|  |  | Maak een lijst met alle spelers die in de jaren 60 geboren zijn.<br>Doe dit met maar één WHERE-conditie (hint: BETWEEN).   | SELECT spelersnr, naam, geb_datum<br>FROM spelers<br>WHERE EXTRACT(YEAR FROM geb_datum) BETWEEN<br>1960 AND 1969  |  |
|  |  | Geef alle spelers die een wedstrijd met twee sets of meer verschil gewonnen of verloren hebben. Zorg voor de juiste output van de gegevens met onder andere vermelding van spelersnaam en eerste cijfer van het spelersnummer tussen haakjes. (gebruik hiervoor een CAST functie om het nummer om te zetten naar een CHAR(1))<br>Sorteer op spelersnr en daarna op wedstrijdnr.<br>Concateneer met   . | SELECT spelers.naam    ' ('    CAST(spelers.spelersnr AS<br>char(1))    ')' AS speler, wedstrijden.wedstrijdnr<br>FROM spelers, wedstrijden<br>WHERE spelers.spelersnr = wedstrijden.spelersnr<br>AND (gewonnen-verloren >= 2 OR verloren-gewonnen >= 2)<br>ORDER BY spelers.spelersnr, wedstrijden.wedstrijdnr;  |  |
|  |  | Geef alle spelers die in januari begonnen zijn met een bestuursfunctie.<br>Sorteer op spelersnr en begindatum (de oorspronkelijke waarde).<br>Belangrijk: de grotere gegevenset heeft wél bestuursleden die niet in januari begonnen zijn.<br><br>Gebruik    ipv concat.   | SELECT spelers.voorletters, spelers.naam,<br>bestuursleden.functie    ' '   <br>to_char(bestuursleden.begin_datum, 'DD/MM/YYYY') AS<br>bestuursfunctie<br>FROM bestuursleden, spelers<br>WHERE bestuursleden.spelersnr = spelers.spelersnr<br>AND EXTRACT(MONTH FROM begin_datum) = 1<br>ORDER BY spelers.spelersnr,<br>bestuursleden.begin_datum;  |  |
|  |  | Geef alle bestuursleden die geboren zijn voor 1970-08-05 en hun staat van dienst.<br>Bekijk de output om de juiste formatering te zien.<br>gebruik een combinatie van CASE, de    functie en datum opmaak.<br>Sorteer op naam en begin datum   | SELECT spelers.voorletters, spelers.naam,<br>bestuursleden.functie, 'actief: '   <br>CASE<br>WHEN bestuursleden.eind_datum IS NULL THEN 'sinds '<br>   to_char(bestuursleden.begin_datum, 'DD/MM/YYYY')<br>ELSE 'van '    to_char(bestuursleden.begin_datum,<br>'DD/MM')    ' tot '    to_char(bestuursleden.eind_datum,<br>'DD/MM/YYYY')<br>END AS periode<br>FROM bestuursleden, spelers<br>WHERE bestuursleden.spelersnr = spelers.spelersnr<br>AND spelers.geb_datum < '1970-08-05'<br>ORDER BY spelers.naam, bestuursleden.begin_datum |  |
|  |  | Geef de geboortedatum van de jongste speler.   | SELECT MAX(geb_datum)<br>FROM spelers;  |  |
|  |  | Geef het gemiddeld aantal verloren sets door vrouwelijke spelers.  | SELECT AVG(verloren)<br>FROM spelers, wedstrijden<br>WHERE spelers.spelersnr = wedstrijden.spelersnr<br>AND spelers.geslacht = 'V';   |  |
|  |  | Geef het grootste boetebedrag voor een speler die momenteel bestuurslid is.<br>Gebruik geen limit.   | SELECT MAX(bedrag)<br>FROM boetes, bestuursleden<br>WHERE boetes.spelersnr = bestuursleden.spelersnr<br>AND bestuursleden.eind_datum IS NULL;   |  |
|  |  | Geef een overzicht van alle spelers met een bondsnummer.   |   |   |










|  |  |   |   |   |
|--|--|---|---|---|
|  |  | <p>Sorteer als volgt:</p> <ul style="list-style-type: none"> <li>- Eerst moeten alle spelers uit zoetermeer getoond worden, deze sorteer je op naam. (spelers met dezelfde naam uit zoetermeer worden gesorteerd op bondsnr)</li> <li>- Daarna de andere spelers, gesorteerd op bondsnr.</li> </ul> <p>Tip: je kan CASE ook gebruiken in de ORDER BY component (zet dan '0')</p>            | <pre>SELECT spelersnr, naam, plaats, bondsnr FROM spelers WHERE bondsnr IS NOT NULL ORDER BY CASE WHEN plaats = 'Zoetermeer' THEN '0' ELSE bondsnr END, naam, bondsnr;</pre>  |    |
|  |  | <p>Geef voor alle bestuursleden een overzicht met hun naam, functie, de leeftijd die ze hadden op het moment van de start van hun functie en de leeftijd die ze hadden op het moment van het einde van hun functie.</p> <p>Tip: gebruik de postgresqlfuncties AGE en EXTRACT om het aantal jaar te krijgen. En CAST(...) as CHAR(2).</p> <p>(Dit is een moeilijke oefening)</p>             | <pre>SELECT spelers.spelersnr, spelers.naam, bestuursleden.functie, CAST(EXTRACT(YEAR FROM AGE(begin_datum, geb_datum)) AS CHAR(2))    ' jaar' AS leeftijd_begin, CASE WHEN eind_datum IS NULL THEN 'nog bezig' ELSE CAST(EXTRACT(YEAR FROM AGE(eind_datum, geb_datum)) AS CHAR(2))    ' jaar' END AS leeftijd_einde FROM spelers, bestuursleden WHERE spelers.spelersnr = bestuursleden.spelersnr;</pre> |    |
|  |  | <p>Geef een lijst met het totaalbedrag aan boetes per speler. Sorteer het hoogste totale boetebedrag bovenaan.</p>  | <pre>SELECT spelersnr, SUM(bedrag) AS totaalboetebedrag FROM boetes GROUP BY spelersnr ORDER BY totaalboetebedrag DESC;</pre>   |    |
|  |  | <p>Geef een lijst met voor elke speler met een boete, het totaal bedrag aan boetes. Als het aantal boetes echter groter is dan 2, zet je 'veel boetes'. Bij de andere spelers zet je een -. Sorteer op aantal boetes aflopend en daarna op totaal boetebedrag aflopend.</p>   | <pre>SELECT spelersnr, SUM(bedrag) AS totaalboetebedrag, CASE WHEN COUNT(bedrag) &gt; 2 THEN 'veel boetes' ELSE '-' END AS status FROM boetes GROUP BY spelersnr ORDER BY COUNT(bedrag) DESC, SUM(bedrag) DESC;</pre>   |    |
|  |  | <p>Geef de langste bestuurstermijn in dagen voor bestuursleden die een wedstrijd gewonnen hebben.</p>   | <pre>SELECT MAX(eind_datum - begin_datum) AS bestuurstermijn FROM bestuursleden, wedstrijden WHERE bestuursleden.spelersnr = wedstrijden.spelersnr AND wedstrijden.gewonnen &gt; wedstrijden.verloren;</pre>  |    |
|  |  | <p>Geef per team, waarvoor wedstrijden gespeeld zijn, het gemiddeld aantal gewonnen en verloren sets. Rond de gemiddelden af tot op twee cijfers na de komma. (gebruik hiervoor de postgresql functie ROUND()) Sorteer op teamnr.</p>   | <pre>SELECT teams.teamnr, teams.divisie, ROUND(AVG(gewonnen),2) AS gewonnen, ROUND(AVG(verloren),2) AS verloren FROM teams, wedstrijden WHERE teams.teamnr = wedstrijden.teamnr GROUP BY teams.teamnr, teams.divisie ORDER BY teams.teamnr;</pre>   |    |
|  |  | <p>Geef het spelersnummer, het jaar van toetreden en bondsnummer van alle spelers en de teams waarvoor ze kapitein zijn. Ook spelers die geen teamkapitein zijn, moeten getoond worden. Sorteer op spelersnr aflopend.</p>  | <pre>SELECT S.spelersnr, S.jaartoe, S.bondsnr, T.teamnr FROM spelers as S LEFT OUTER JOIN teams as T ON S.spelersnr = T.spelersnr ORDER BY S.spelersnr DESC</pre>   |  |
|  |  | <p>Geef voor alle spelers die bestuurslid zijn (of geweest zijn) een lijst van de wedstrijdnummers van alle wedstrijden die ze gespeeld hebben, en het verschil in sets waarmee ze gewonnen hebben. Bestuursleden zonder wedstrijd moeten ook in het resultaat voorkomen. Sorteer op bestuurslidfunctie (oplopend), verschil (aflopend), spelersnr (oplopend) en wedstrijdnr (oplopend)</p> | <pre>SELECT S.spelersnr, S.naam, B.functie, W.wedstrijdnr, gewonnen - verloren as verschil FROM spelers S INNER JOIN bestuursleden B ON S.spelersnr = B.spelersnr LEFT OUTER JOIN wedstrijden W ON S.spelersnr = W.spelersnr ORDER BY B.functie ASC, verschil DESC, S.spelersnr ASC, W.wedstrijdnr ASC;</pre>   |  |
|  |  | <p>Geef voor alle vrouwelijke spelers die in Den Haag, Zoetermeer of Leiden wonen het spelersnummer, hun woonplaats en een lijst van de teams waarvoor ze ooit gespeeld hebben. Sorteer op spelersnr</p>  | <pre>SELECT S.spelersnr, plaats, teamnr FROM spelers S LEFT OUTER JOIN wedstrijden W ON S.spelersnr = W.spelersnr WHERE geslacht = 'V' AND plaats IN ('Den Haag', 'Zoetermeer', 'Leiden') ORDER BY S.spelersnr</pre>  |  |
|  |  | <p>Geef voor elke wedstrijd het wedstrijdnummer en de volledige naam van de aanvoerder van het team waarvoor de wedstrijd werd gespeeld. Sorteer je resultaat volgens het wedstrijdnummer in oplopende volgorde.</p>  | <pre>SELECT wedstrijdnr, naam, voorletters FROM (wedstrijden W INNER JOIN teams T ON W.teamnr = T.teamnr) INNER JOIN spelers S ON T.spelersnr = S.spelersnr ORDER BY 1</pre>  |  |
|  |  | <p>Geef voor elke mannelijke speler wiens naam minstens 2 keer de letter 'e' bevat een lijst van de functies die hij op dit moment uitoefent. Ook mannelijke spelers zonder huidige functie moeten getoond worden. Sorteer op spelersnr.</p>  | <pre>select naam, geslacht, functie from spelers left outer join bestuursleden on spelers.spelersnr = bestuursleden.spelersnr and bestuursleden.eind_datum is null where geslacht = 'M' and naam like '%e%e%' ORDER BY spelers.spelersnr;</pre>   |  |
|  |  | <p>Geef alle spelers die meer dan één wedstrijd gewonnen hebben.</p>  | <pre>SELECT spelers.spelersnr, spelers.naam FROM wedstrijden, spelers WHERE wedstrijden.spelersnr = spelers.spelersnr AND wedstrijden.gewonnen &gt; wedstrijden.verloren GROUP BY spelers.spelersnr, spelers.naam HAVING COUNT(*) &gt; 1</pre>  |  |
|  |  | <p>Geef voor alle actieve bestuursleden die al een boete gehad hebben van meer dan 50 euro hun maximale boetebedrag. Gebruik geen DISTINCT.</p>   | <pre>SELECT bestuursleden.spelersnr, MAX(boetes.bedrag) AS maximum FROM boetes, bestuursleden WHERE boetes.spelersnr = bestuursleden.spelersnr AND bestuursleden.eind_datum IS NULL GROUP BY bestuursleden.spelersnr HAVING MAX(boetes.bedrag) &gt; 50;</pre>   |  |
|  |  | <p>Sorteer de teams in functie van het aantal verloren wedstrijden (oplopend). Als het aantal verloren wedstrijden gelijk is, sorteer je op het totaal aantal gewonnen sets (aflopend) en op divisienaam (oplopend). Een wedstrijd die verloren werd met 3 - 0 telt niet mee in deze output.</p>  | <pre>SELECT teams.divisie, COUNT(*) AS aantal FROM wedstrijden, teams WHERE wedstrijden.teamnr = teams.teamnr AND wedstrijden.verloren &gt; wedstrijden.gewonnen AND NOT (wedstrijden.verloren = 3 AND wedstrijden.gewonnen = 0)</pre>  |  |













|  |  |  |  |   |
|--|--|--|--|---|
|  |  | Geef het aantal wedstrijden dat voldoet aan bovenstaande voorwaarde mee in de output.  | GROUP BY teams.divisie<br>ORDER BY aantal, SUM(wedstrijden.gewonnen) DESC,<br>teams.divisie;   |   |
|  |  | Geef de teams en het aantal verschillende spelers dat voor dit team gespeeld heeft.<br>Sorteer op divisie.   | SELECT teams.divisie, COUNT(DISTINCT<br>wedstrijden.spelersnr) AS aantal<br>FROM wedstrijden, teams<br>WHERE wedstrijden.teamnr = teams.teamnr<br>GROUP BY teams.divisie<br>ORDER BY teams.divisie;  |    |
|  |  | Geef het aantal spelers met een bondsnummer die een boete gehad hebben.  | SELECT COUNT(DISTINCT spelers.bondsnr) AS aantal<br>FROM spelers, boetes<br>WHERE spelers.spelersnr = boetes.spelersnr;  |    |
|  |  | Geef de teams (het nummer van het team en de divisie) waarvoor meer dan vier verschillende spelers gespeeld hebben.  | SELECT teams.teamnr, teams.divisie<br>FROM wedstrijden, teams<br>WHERE wedstrijden.teamnr = teams.teamnr<br>GROUP BY teams.teamnr, teams.divisie<br>HAVING COUNT(DISTINCT wedstrijden.spelersnr) > 4;  |    |
|  |  | Maak een lijst met per spelersnummer het aantal wedstrijden dat een speler verloren heeft op voorwaarde dat deze speler meer dan een wedstrijd verloren heeft. Bijkomend moet de lijst gesorteerd worden op basis van het spelersnummer.   | SELECT spelersnr, COUNT(*) AS aantal<br>FROM wedstrijden<br>WHERE gewonnen < verloren<br>GROUP BY spelersnr<br>HAVING COUNT(*) > 1<br>ORDER BY spelersnr;  |    |
|  |  | Geef een lijst met de spelers die ooit bestuurslid zijn geweest (of nog steeds zijn) en niet in Den Haag of Zoetermeer wonen. Bijkomend mag deze speler maximaal 2 keer in het bestuur van de club gezeteld hebben.<br>De lijst moet aflopend gesorteerd worden op het aantal maal dat de betreffende speler in het bestuur zetelde. Mensen met hetzelfde aantal keren moeten oplopend gesorteerd worden op basis van hun spelersnr. | SELECT spelers. naam, COUNT(*) AS aantal<br>FROM spelers, bestuursleden<br>WHERE bestuursleden.spelersnr = spelers.spelersnr<br>AND plaats NOT IN ('Den Haag', 'Zoetermeer')<br>GROUP BY spelers.spelersnr, spelers. naam<br>HAVING COUNT(*) <= 2<br>ORDER BY aantal DESC, spelers.spelersnr;  |    |
|  |  | Geef het gemiddeld aantal gewonnen en verloren sets per geboortjaar.<br>Sorteer op geboortjaar.  | SELECT EXTRACT(YEAR FROM geb_datum) AS<br>geboortjaar, AVG(gewonnen) AS gewonnen,<br>AVG(verloren) AS verloren<br>FROM wedstrijden, spelers<br>WHERE wedstrijden.spelersnr = spelers.spelersnr<br>GROUP BY EXTRACT(YEAR FROM geb_datum)<br>ORDER BY geboortjaar;   |    |
|  |  | Geef spelers die in het jaar dat ze lid geworden zijn van de club reeds een boete van meer dan 50 euro gekregen hebben en de som van al deze boetes groter of gelijk is aan 100 euro.<br>Geef buiten de voorletters en de naam van de speler ook het aantal boetes die aan bovenstaande voorwaarden voldoen.<br>Sorteer op spelersnr.  | SELECT spelers.voorletters, spelers. naam, COUNT(*) AS<br>aantalboetes<br>FROM spelers, boetes<br>WHERE spelers.spelersnr = boetes.spelersnr<br>AND boetes.bedrag > 50<br>AND EXTRACT(YEAR FROM boetes.datum) =<br>spelers.jaartoe<br>GROUP BY spelers.spelersnr, spelers.voorletters,<br>spelers. naam<br>HAVING SUM (bedrag) >= 100<br>ORDER BY spelers.spelersnr;                                   |  |
|  |  | Geef de team(s) waarbij de standaardafwijking van de verloren sets kleiner is dan de standaardafwijking van de gewonnen sets.<br>Sorteer op teamnr   | SELECT teamnr<br>FROM wedstrijden<br>GROUP BY teamnr<br>HAVING STDDEV(verloren) < STDDEV(gewonnen)<br>ORDER BY teamnr;   |  |
|  |  | Geef een alfabetisch gesorteerde lijst van de namen van alle leden van de tennisvereniging die ofwel een recreatiespeler zijn (speelt geen wedstrijden) ofwel een wedstrijdspeler die nog geen wedstrijden gespeeld heeft.   | SELECT naam<br>FROM spelers S left outer join wedstrijden W<br>ON S.spelersnr = W.spelersnr<br>WHERE W.wedstrijdnr is null<br>ORDER BY naam;   |  |
|  |  | Geef voor alle huidige bestuursleden hun functie en de lijst van boetes die voor hen werd betaald.<br>Omdat je dit wil vergelijken met de boetebedragen die betaald werden voor spelers die niet in het bestuur zitten, wil je deze boetebedragen ook opnemen in de tweede kolom van je resultaat.<br>Sorteer je antwoord eerst op functie en daarna op het boetebedrag.   | SELECT functie, bedrag<br>FROM boetes BT FULL OUTER JOIN bestuursleden B<br>ON (BT.spelersnr = B.spelersnr<br>AND eind_datum is null)<br>ORDER BY 1,2;   |  |
|  |  | Geef een aflopend gesorteerde lijst van de nummers van alle spelers waarvoor nog geen boete werd betaald en die nog nooit in het bestuur van de tennisvereniging hebben gezeten.   | SELECT S.spelersnr<br>FROM spelers S LEFT OUTER JOIN boetes BT<br>ON S.spelersnr = BT.spelersnr<br>LEFT OUTER JOIN bestuursleden B<br>ON S.spelersnr = B.spelersnr<br>WHERE BT.spelersnr is null<br>AND B.spelersnr is null<br>ORDER BY 1 DESC;  |  |
|  |  | Geef alle spelers die geen boete gekregen hebben en niet in Den Haag wonen.<br>Sorteer op jaar van toetreden.  | SELECT spelers.spelersnr, naam, plaats<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr =<br>boetes.spelersnr<br>WHERE boetes.betalingsnr IS NULL<br>AND spelers.plaats <> 'Den Haag'<br>ORDER BY jaartoe;   |  |
|  |  | Geef per klant het totaal aantal reizen waaraan deze klant zal deelnemen en het langste bezoek dat deze klant zal maken aan een hemelobject, over alle reizen heen.<br>Klanten zonder reizen of zonder bezoeken moeten ook voorkomen in het overzicht.<br>Sorteer op klantnr.  | SELECT klanten.klantnr, COUNT(DISTINCT reizen.reisnr)<br>AS aantal, MAX(bezoeken.verblijfsduur) AS langstebezoek<br>FROM klanten<br>LEFT OUTER JOIN deelnames ON klanten.klantnr =<br>deelnames.klantnr<br>LEFT OUTER JOIN reizen ON deelnames.reisnr =<br>reizen.reisnr<br>LEFT OUTER JOIN bezoeken ON deelnames.reisnr =<br>bezoeken.reisnr<br>GROUP BY klanten.klantnr<br>ORDER BY klanten.klantnr; |  |
|  |  |  | SELECT teams.teamnr, teams.divisie,<br>wedstrijden.wedstrijdnr, wedstrijden.spelersnr,<br>CASE<br>WHEN bestuursleden.spelersnr IS NULL THEN ''   |   |

|  |  |  |   |   |
|--|--|--|---|---|
|  |  | Geef per team de verloren wedstrijden. Zorg dat teams zonder verloren wedstrijden ook in de output verschijnen.<br>Duid per wedstrijd aan of het om een actief bestuurslid gaat.<br>Sorteer op divisie en wedstrijdnummer.   | ELSE 'actief'<br>END AS bestuurslid<br>FROM teams<br>LEFT OUTER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr AND wedstrijden.verloren > wedstrijden.gewonnen<br>LEFT OUTER JOIN bestuursleden ON wedstrijden.spelersnr = bestuursleden.spelersnr AND bestuursleden.eind_datum IS NULL<br>ORDER BY divisie, wedstrijden.wedstrijdnr;  |    |
|  |  | Geef van elke speler wonend in Rijswijk het spelersnr, de naam, de lijst met boetebedragen (indien er zijn) en de lijst met teams waarvoor hij/zij een wedstrijd gespeeld heeft (indien hij er gespeeld heeft). Geef het resultaat volgens oplopend spelersnr, teamnr en bedrag                                  | SELECT s.spelersnr, s.naam, bedrag, teamnr<br>FROM ((spelers s LEFT OUTER JOIN boetes b ON s.spelersnr = b.spelersnr)<br>LEFT OUTER JOIN wedstrijden w ON w.spelersnr = s.spelersnr)<br>WHERE plaats = 'Rijswijk'<br>ORDER BY s.spelersnr, w.teamnr, bedrag   |    |
|  |  | Geef een lijst van alle reizen met tenminste één bezoek aan de Maan of aan Mars. De lijst moet stijgend gesorteerd zijn op basis van het vertrekdatum.   | SELECT DISTINCT bezoeken.reisnr, reizen.vertrekdatum<br>FROM bezoeken<br>INNER JOIN reizen ON bezoeken.reisnr = reizen.reisnr<br>WHERE objectnaam IN ('Maan', 'Mars')<br>ORDER BY vertrekdatum;   |    |
|  |  | Welke reizigers hebben al meer dan 1 reis ondernomen waarvoor ze meer dan 2,5 miljoen euro moesten betalen?<br>Sorteer op naam   | SELECT klanten.naam, COUNT(*) AS aantal_reizen<br>FROM reizen<br>INNER JOIN deelnames ON reizen.reisnr = deelnames.reisnr<br>INNER JOIN klanten ON deelnames.klantnr = klanten.klantnr<br>WHERE reizen.prijs > 2.50<br>GROUP BY klanten.klantnr, klanten.naam<br>HAVING COUNT(*) > 1<br>ORDER BY naam;  |    |
|  |  | Welke klanten (klantnaam, geboortedatum) zijn op een ruimtereis vertrokken in het jaar dat ze 45 jaar geworden zijn?<br>Het resultaat moet stijgend gesorteerd worden op de geboortedatum.   | SELECT DISTINCT naam AS klantnaam, klanten.geboortedatum<br>FROM klanten<br>INNER JOIN deelnames ON klanten.klantnr = deelnames.klantnr<br>INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br>WHERE EXTRACT(YEAR FROM reizen.vertrekdatum) - EXTRACT(YEAR FROM klanten.geboortedatum) = 45<br>ORDER BY klanten.geboortedatum;   |    |
|  |  | Bereken voor de klant wiens naam begint met G en eindigt met s hoeveel hij in totaal al besteed heeft aan reizen.  | SELECT klanten.naam, SUM(prijs)<br>FROM reizen<br>INNER JOIN deelnames ON reizen.reisnr = deelnames.reisnr<br>INNER JOIN klanten ON deelnames.klantnr = klanten.klantnr<br>WHERE klanten.naam LIKE 'G%s'<br>GROUP BY klanten.naam;  |  |
|  |  | Maak een lijst met een overzicht van de reizen en het aantal deelnemers van elke reis. Orden de lijst dalend op basis van het aantal deelnemers, als er eenzelfde aantal deelnemers is, moeten deze stijgend geordend worden volgens reisnummer.   | SELECT reizen.reisnr, COUNT(*) AS deelnemers<br>FROM reizen<br>INNER JOIN deelnames ON reizen.reisnr = deelnames.reisnr<br>GROUP BY reizen.reisnr<br>ORDER BY deelnemers DESC, reisnr;  |  |
|  |  | Welke reizen hebben exact drie verschillende hemelobjecten als reisdoel?<br>Sorteer op reisnr.   | SELECT reisnr<br>FROM bezoeken<br>GROUP BY reisnr<br>HAVING COUNT(DISTINCT objectnaam) = 3<br>ORDER BY reisnr;  |  |
|  |  | Maak een lijst met een overzicht van de reizen en het aantal deelnemers van elke reis. Orden de lijst dalend op basis van het aantal deelnemers, als er eenzelfde aantal deelnemers is, moeten deze stijgend geordend worden volgens reisnummer.<br>Zorg dat reizen zonder deelnames ook in het resultaat staan. | SELECT reizen.reisnr, COUNT(deelnames.reisnr) AS deelnemers<br>FROM reizen<br>LEFT OUTER JOIN deelnames ON reizen.reisnr = deelnames.reisnr<br>GROUP BY reizen.reisnr<br>ORDER BY deelnemers DESC, reisnr;  |  |
|  |  | Geef een lijst met alle planeten en per planeet zijn satellieten.<br>Sorteer op planeet en daarna op satelliet.  | SELECT planeten.objectnaam as planeet, satellieten.objectnaam AS maan<br>FROM hemelobjecten planeten<br>LEFT OUTER JOIN hemelobjecten satellieten ON satellieten.satellietvan = planeten.objectnaam<br>WHERE planeten.satellietvan = 'Zon'<br>ORDER BY planeet, maan;   |  |
|  |  | Op welke planeten verblijft men gemiddeld langer dan 2 dagen?<br>Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam, AVG(verblijfsduur)<br>FROM bezoeken<br>INNER JOIN hemelobjecten ON bezoeken.objectnaam = hemelobjecten.objectnaam<br>WHERE hemelobjecten.satellietvan = 'Zon'<br>GROUP BY hemelobjecten.objectnaam<br>HAVING AVG(verblijfsduur) > 2<br>ORDER BY hemelobjecten.objectnaam;  |  |
|  |  | Welke planeten met meer dan 7 manen worden bezocht (met of zonder verblijf)?<br>Sorteer aflopend op basis van het aantal manen.<br>Let erop dat je planeten die meerdere keren bezocht worden, niet dubbel telt.   | SELECT planeten.objectnaam<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten satellieten ON satellieten.satellietvan = planeten.objectnaam<br>INNER JOIN hemelobjecten sterren ON planeten.satellietvan = sterren.objectnaam<br>INNER JOIN bezoeken ON planeten.objectnaam = bezoeken.objectnaam<br>WHERE sterren.satellietvan IS NULL<br>GROUP BY planeten.objectnaam<br>HAVING COUNT(DISTINCT satellieten.objectnaam) > 7<br>ORDER BY COUNT(DISTINCT satellieten.objectnaam) DESC; |  |
|  |  | Bereken voor alle hemelobjecten die satellieten hebben, het aantal   | SELECT planeten.objectnaam, COUNT(*)<br>FROM hemelobjecten planeten   |   |















































|  |  |  |   |   |
|--|--|--|---|---|
|  |  | satellieten per hemelobject.<br>De lijst moet dalend gesorteerd worden op basis van het aantal satellieten van de hemelobjecten en daarna alfabetisch op basis van objectnaam.                                   | INNER JOIN hemelobjecten satellieten ON satellieten.satellietvan = planeten.objectnaam<br>GROUP BY planeten.objectnaam<br>ORDER BY COUNT(*) DESC, planeten.objectnaam;  |     |
|  |  | Geef het kleinste reisnummer met een bezoek aan de maan.   | SELECT MIN(reisnr)<br>FROM bezoeken<br>WHERE objectnaam = 'Maan';   |    |
|  |  | Geef de reizen met meer dan drie klanten.<br>Sorteer op reisnr.  | SELECT deelnames.reisnr<br>FROM deelnames<br>INNER JOIN bezoeken ON deelnames.reisnr = bezoeken.reisnr<br>LEFT OUTER JOIN satellieten ON satellieten.satellietvan = planeten.objectnaam<br>GROUP BY deelnames.reisnr<br>HAVING count(distinct klantnr) > 3<br>ORDER BY deelnames.reisnr   |    |
|  |  | Geef alle planeten (gesorteerd op afstand van de zon) en het aantal verschillende reizen die deze planeet bezocht hebben.  | SELECT hemelobjecten.objectnaam, COUNT(DISTINCT bezoeken.reisnr) AS aantalbezoeken<br>FROM hemelobjecten<br>LEFT OUTER JOIN bezoeken ON hemelobjecten.objectnaam = bezoeken.objectnaam<br>WHERE satellietvan = 'Zon'<br>GROUP BY hemelobjecten.afstand, hemelobjecten.objectnaam<br>ORDER BY afstand;   |    |
|  |  | Geef per klant het totaal aantal dagen dat deze klant in totaal op een planeet zal verblijven.<br>Sorteer op klantnr.  | SELECT klanten.klantnr, SUM(bezoeken.verblijfsduur) AS totaleverblijfsduur<br>FROM klanten<br>INNER JOIN deelnames ON klanten.klantnr = deelnames.klantnr<br>INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br>INNER JOIN bezoeken ON reizen.reisnr = bezoeken.reisnr<br>INNER JOIN hemelobjecten P ON bezoeken.objectnaam = P.objectnaam<br>INNER JOIN hemelobjecten S ON (P.satellietvan = S.objectnaam AND S.satellietvan IS NULL)<br>GROUP BY klanten.klantnr<br>ORDER BY klantnr; |    |
|  |  | Sorteer de klanten aflopend op gemiddelde kostprijs per bezoek (totaalprijs van alle reizen per klant/totaal aantal dagen verblijf op een hemelobject), op twee cijfers na de komma afgerond, daarna op klantnr. | SELECT klanten.klantnr, ROUND(SUM(reizen.prijs)/SUM(bezoeken.verblijfsduur),2) AS gemiddeldeprijs<br>FROM klanten<br>INNER JOIN deelnames ON klanten.klantnr = deelnames.klantnr<br>INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br>INNER JOIN bezoeken ON reizen.reisnr = bezoeken.reisnr<br>GROUP BY klanten.klantnr<br>ORDER BY gemiddeldeprijs DESC, klantnr;  |    |
|  |  | Geef de hemelobjecten die nog nooit bezocht zijn (een bezoek heeft een verblijfsduur van minstens één dag).  | SELECT hemelobjecten.objectnaam<br>FROM hemelobjecten<br>LEFT OUTER JOIN bezoeken ON hemelobjecten.objectnaam = bezoeken.objectnaam AND bezoeken.verblijfsduur > 0<br>WHERE bezoeken.objectnaam IS NULL<br>ORDER BY hemelobjecten.objectnaam;   |  |
|  |  | Geef de reizen waar klanten geboren in 1980 aan deelgenomen hebben en die de maan bezocht hebben (met of zonder verblijfsduur). Vermeld ook de volgnummers van alle bezoeken.<br>Sorteer op reisnr en op volgnr. | SELECT klanten.klantnr, reizen.reisnr, bezoeken.volgnr<br>FROM klanten<br>INNER JOIN deelnames ON klanten.klantnr = deelnames.klantnr<br>INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br>INNER JOIN bezoeken ON reizen.reisnr = bezoeken.reisnr<br>WHERE bezoeken.objectnaam = 'Maan'<br>AND EXTRACT(YEAR FROM klanten.geboortedatum) = 1980<br>ORDER BY reizen.reisnr, bezoeken.volgnr;   |  |
|  |  | Geef de planeten en het aantal verschillende personen die hen bezocht hebben (met of zonder verblijf).<br>Sorteer op objectnaam.   | SELECT hemelobjecten.objectnaam, COUNT(DISTINCT deelnames.klantnr) AS aantalbezoekers<br>FROM hemelobjecten<br>LEFT OUTER JOIN bezoeken ON hemelobjecten.objectnaam = bezoeken.objectnaam<br>LEFT OUTER JOIN deelnames ON bezoeken.reisnr = deelnames.reisnr<br>WHERE hemelobjecten.satellietvan = 'Zon'<br>GROUP BY hemelobjecten.objectnaam<br>ORDER BY hemelobjecten.objectnaam;   |  |
|  |  | Geef de grootste diameter van een planeet die bezocht is (met of zonder verblijf) door iemand met een 'a' in zijn of haar naam.  | SELECT MAX(diameter) AS grootste<br>FROM hemelobjecten<br>INNER JOIN bezoeken ON hemelobjecten.objectnaam = bezoeken.objectnaam<br>INNER JOIN deelnames ON bezoeken.reisnr = deelnames.reisnr<br>INNER JOIN klanten ON deelnames.klantnr = klanten.klantnr<br>WHERE hemelobjecten.satellietvan = 'Zon'<br>AND klanten. naam like '%a%';   |  |
|  |  | Geef de datum van de eerste reis met een bezoek aan Io.  | SELECT TO_CHAR(MIN(reizen. vertrekdatum), 'dd/mm/yyyy') AS vertrekdatum<br>FROM reizen<br>INNER JOIN bezoeken ON reizen.reisnr = bezoeken.reisnr<br>WHERE bezoeken.objectnaam = 'Io';   |  |
|  |  | Geef voor de actieve bestuursleden zonder boete hun laatste  | SELECT bestuursleden.spelersnr, MAX(wedstrijden.wedstrijdnr) AS laatstewedstrijd<br>FROM bestuursleden<br>INNER JOIN wedstrijden ON bestuursleden.spelersnr = wedstrijden.spelersnr AND bestuursleden.eind_datum IS   |   |

|  |  |   |   |   |
|--|--|---|---|---|
|  |  | gespeelde wedstrijd (die met het hoogste wedstrijdnummer).<br>Sorteer aflopend op spelersnr.  | NULL<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr = boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY bestuursleden.spelersnr<br>ORDER BY spelersnr DESC;  |     |
|  |  | Geef het gemiddeld boetebedrag per speler, afgerond op twee cijfers na de komma. spelers zonder boete moeten bovenaan staan met 'geen boetes'<br>Sorteer daarna op spelersnaam en gemiddeld boetebedrag.<br>Om een waarde om te zetten naar een ander datatype, kan je ofwel CAST() gebruiken ofwel de TO_CHAR() met 'FM999.00' als opmaakmasker.         | SELECT spelers.naam,<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 'geen boetes'<br>ELSE CAST(ROUND(AVG(boetes.bedrag), 2) AS varchar(8))<br>END<br>AS gemiddeld<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr = boetes.spelersnr<br>GROUP BY spelers.spelersnr, spelers.naam<br>ORDER BY<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 0<br>ELSE 1<br>END, spelers.naam, AVG(boetes.bedrag);  |    |
|  |  | Geef per team de leeftijd van de aanvoerder (tip: postgresql heeft een AGE() functie) en het aantal verschillende spelers dat voor dit team gespeeld heeft.<br>Alleen teams waarvoor wedstrijden zijn gespeeld en die een aanvoerder hebben, moeten vermeld worden.<br>Sorteer op leeftijd en daarna op aantal verschillende spelers en daarna op teamnr. | SELECT teams.teamnr, CAST(EXTRACT(year from AGE(NOW(), spelers.geb_datum)) AS varchar(2))    ' jaar'<br>AS leeftijd, COUNT(DISTINCT wedstrijden.spelersnr) AS aantalspelers<br>FROM teams<br>INNER JOIN spelers ON teams.spelersnr = spelers.spelersnr<br>INNER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr<br>GROUP BY teams.teamnr, spelers.geb_datum<br>ORDER BY leeftijd, aantalspelers, teamnr;  |    |
|  |  | Geef het hoogste wedstrijdnummer voor de teams waarvoor wedstrijden gespeeld zijn door bestuursleden (actief en niet meer actief) die geen boete hebben gekregen.<br>Sorteer op teamnr.   | SELECT teams.teamnr, MAX(wedstrijden.wedstrijdnr) AS laatstewedstrijd<br>FROM teams<br>INNER JOIN wedstrijden on teams.teamnr = wedstrijden.teamnr<br>INNER JOIN bestuursleden ON wedstrijden.spelersnr = bestuursleden.spelersnr<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr = boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY teams.teamnr<br>ORDER BY teams.teamnr;  |    |
|  |  | Geef alle spelers die meer wedstrijden gespeeld hebben dan het aantal wedstrijden dat de huidige voorzitter heeft verloren. De huidige voorzitter komt zelf niet in de lijst voor. Gebruik geen subqueries.<br>Sorteer op spelersnr.  | SELECT spelers.spelersnr<br>FROM spelers<br>INNER JOIN wedstrijden ON spelers.spelersnr = wedstrijden.spelersnr,<br>bestuursleden<br>INNER JOIN wedstrijden voorzitterswedstrijden ON bestuursleden.spelersnr = voorzitterswedstrijden.spelersnr<br>AND voorzitterswedstrijden.gewonnen < voorzitterswedstrijden.verloren<br>WHERE bestuursleden.eind_datum IS NULL AND bestuursleden.functie = 'Voorzitter' AND bestuursleden.spelersnr <> spelers.spelersnr<br>GROUP BY spelers.spelersnr<br>HAVING COUNT(DISTINCT wedstrijden.wedstrijdnr) > COUNT(DISTINCT voorzitterswedstrijden.wedstrijdnr)<br>ORDER BY spelers.spelersnr; |  |
|  |  | Geef de leeftijd van de jongste klant op moment van vertrek.  | SELECT EXTRACT (YEAR FROM MIN(AGE(vertrekdatum, geboortedatum))) AS jongsteleeftijd<br>FROM klanten<br>INNER JOIN deelnames ON klanten.klantnr = deelnames.klantnr<br>INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr;  |  |
|  |  | Geef de diameter van de grootste, niet bezochte maan (satelliet van een planeet).   | SELECT MAX(manen.diameter) AS grootstemaan<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten manen ON planeten.objectnaam = manen.satellietvan AND planeten.satellietvan = 'Zon'<br>LEFT OUTER JOIN bezoeken ON manen.objectnaam = bezoeken.objectnaam<br>WHERE bezoeken.reisnr IS NULL;   |  |
|  |  | Geef de naam, diameter en het langste verblijf op alle hemelobjecten met minder dan 5 hemelobjecten die rond dit object draaien. Geef enkel hemelobjecten die bezocht zijn of waar gepasseerd wordt.<br>Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam,<br>hemelobjecten.diameter, MAX(bezoeken.verblijfsduur) AS maximale_verblijf<br>FROM bezoeken<br>INNER JOIN hemelobjecten ON bezoeken.objectnaam = hemelobjecten.objectnaam<br>LEFT OUTER JOIN hemelobjecten manen ON hemelobjecten.objectnaam = manen.satellietvan<br>GROUP BY hemelobjecten.objectnaam,<br>hemelobjecten.diameter<br>HAVING COUNT(DISTINCT manen.objectnaam) < 5<br>ORDER BY objectnaam;  |  |
|  |  | Geef voor elk hemelobject de minimale en maximale gemiddelde afstand tot zijn zon (de centrale ster in een sterrenstelsel) als je weet dat de kolom 'afstand' de gemiddelde afstand bevat tot het hemelobject waar rond ze draaien. Met de grootte van het hemelobject hoef je geen rekening te houden.<br>Sorteer op minimale afstand en op objectnaam.  | SELECT hemelobjecten.objectnaam,<br>CASE<br>WHEN hemelobjecten.afstand IS NULL THEN 0<br>WHEN draaitrond.afstand IS NULL THEN hemelobjecten.afstand<br>ELSE draaitrond.afstand + hemelobjecten.afstand<br>END AS maximale_afstand,<br>CASE<br>WHEN hemelobjecten.afstand IS NULL THEN 0<br>WHEN draaitrond.afstand IS NULL THEN hemelobjecten.afstand<br>ELSE draaitrond.afstand - hemelobjecten.afstand<br>END AS minimale_afstand   |  |
















|  |  |  |  |   |
|--|--|--|--|---|
|  |  |  | FROM hemelobjecten<br>LEFT OUTER JOIN hemelobjecten draaitrond ON<br>hemelobjecten.satellietvan = draaitrond.objectnaam<br>ORDER BY minimale_afstand, objectnaam;  |   |
|  |  | Geef het spelersnummer en bondsnummer van alle spelers die<br>jonger zijn dan de speler met bondsnummer 8467.<br>Sorteer op spelersnr  | SELECT S.spelersnr, S.bondsnr<br>FROM spelers as S INNER JOIN spelers as P<br>ON S.geb_datum > P.geb_datum<br>WHERE P.bondsnr = '8467'<br>ORDER BY S.spelersnr;  |    |
|  |  | Geef een lijst van alle spelers die nog geen wedstrijd gespeeld<br>hebben.<br>Sorteer op spelersnr   | SELECT spelers.spelersnr, naam<br>FROM spelers<br>LEFT OUTER JOIN wedstrijden ON spelers.spelersnr =<br>wedstrijden.spelersnr<br>WHERE wedstrijden.wedstrijdnr IS NULL<br>ORDER BY spelers.spelersnr;  |    |
|  |  | Geef een lijst met spelers (naam, voorletters en geboortedatum)<br>die nog geen boete gekregen hebben met een bedrag hoger dan<br>75 euro.<br>Sorteer op spelersnr.  | SELECT spelers.naam, spelers.voorletters,<br>TO_CHAR(spelers.geb_datum, 'DD/MM/YYYY') AS<br>geboortedatum<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr =<br>boetes.spelersnr AND boetes.bedrag > 75<br>WHERE boetes.betalingsnr IS NULL<br>ORDER BY spelers.spelersnr;   |    |
|  |  | Geef een lijst van alle huidige bestuursleden die nog geen boete<br>gekregen hebben, maar wel al minstens één wedstrijd verloren<br>hebben.<br>Sorteer op spelersnr en wedstrijdnr.  | SELECT S.spelersnr, S.naam, B.functie, B.begin_datum,<br>W.wedstrijdnr<br>FROM spelers S<br>INNER JOIN bestuursleden B ON S.spelersnr =<br>B.spelersnr AND B.eind_datum IS NULL<br>INNER JOIN wedstrijden W ON S.spelersnr = W.spelersnr<br>AND W.gewonnen < W.verloren<br>LEFT OUTER JOIN boetes BO ON S.spelersnr =<br>BO.spelersnr<br>WHERE BO.betalingsnr IS NULL<br>ORDER BY S.spelersnr, W.wedstrijdnr   |    |
|  |  | Geef een overzicht van alle boetes, onderverdeeld in categorie:<br><br>LAAG: boetebedrag tussen 0 en 40<br>MIDDELMATIG: tussen 40 en 80<br>HOOG: meer dan 80<br><br>In overzicht toon je betalingsnr, spelersnr, spelersnaam, bedrag en<br>categorie van de boete<br><br>Sorteer op naam van de speler   | select naam, spelers.spelersnr, betalingsnr, bedrag,<br>case when bedrag < 40 then 'laag'<br>when bedrag < 80 then 'middelmatig'<br>else 'hoog'<br>end as categorie<br>from spelers, boetes<br>where spelers.spelersnr = boetes.spelersnr<br>order by naam   |    |
|  |  | Reken uit hoeveel het zou kosten voor een deelnemer om deel te<br>nemen aan alle beschikbare ruimtereizen.   | SELECT SUM(prijs)<br>FROM reizen   |  |
|  |  | Tel per reis het aantal bezochte hemelobjecten. Sorteer op reisnr.   | SELECT reisnr, COUNT(reisnr)<br>FROM bezoeken<br>GROUP BY reisnr<br>ORDER BY reisnr  |  |
|  |  | Zoek de hemelobjecten op die meer dan eens in een zelfde<br>ruimtereis bezocht worden. Geef reisnr, objectnaam en aantal<br>bezoeken. Sorteer op reisnr.   | SELECT reisnr, objectnaam, COUNT(reisnr)<br>FROM bezoeken<br>GROUP BY reisnr, objectnaam<br>HAVING COUNT(reisnr) > 1<br>ORDER BY reisnr  |  |
|  |  | Tel per klant het aantal reizen. Sorteer op naam.  | SELECT klanten.naam, COUNT(deelnames.reisnr)<br>FROM klanten<br>INNER JOIN deelnames USING (klientnr)<br>GROUP BY klanten.naam<br>ORDER BY klanten.naam  |  |
|  |  | Reken per planeet uit wat de gemiddelde afstand van zijn manen<br>is.  | SELECT planeet.objectnaam as planeet,<br>AVG(maan.afstand) as afstand<br>FROM hemelobjecten ster<br>INNER JOIN hemelobjecten planeet ON ster.satellietvan<br>IS NULL AND planeet.satellietvan = ster.objectnaam<br>INNER JOIN hemelobjecten maan ON maan.satellietvan =<br>planeet.objectnaam<br>GROUP BY planeet.objectnaam<br>ORDER BY afstand   |  |
|  |  | Vind per planeet die zijn maan die het verst ervan verwijderd is.<br>Geef planeet, maan en afstand en sorteer stijgend op afstand.   | SELECT planeet.objectnaam as planeet, M.objectnaam as<br>maan, M.afstand as afstand<br>FROM hemelobjecten ster<br>INNER JOIN hemelobjecten planeet ON ster.satellietvan<br>IS NULL AND planeet.satellietvan = ster.objectnaam<br>INNER JOIN hemelobjecten maan ON maan.satellietvan =<br>planeet.objectnaam<br>INNER JOIN hemelobjecten M on M.satellietvan =<br>planeet.objectnaam<br>GROUP BY planeet.objectnaam, M.objectnaam<br>HAVING M.afstand = MAX(maan.afstand)<br>ORDER BY afstand |  |
|  |  | Geef een lijst van alle planeten in het heelal die minstens 1 satelliet<br>hebben met een diamemeter groter dan 2000km (je weet dus niet<br>hoeveel satellieten, kan ook bv 100 zijn, het aantal satelieten<br>bestaat maximaal uit 3 cijfers, gebruik dit bij je omzetting). Voor<br>deze planeten toon je het aantal dergelijke satellieten. Geef ook<br>een lijst van de andere planeten. Bij hen zet je als output: "Geen".<br>Sorteer op planeetnaam. | SELECT planeet.objectnaam, CASE WHEN<br>count(sat.objectnaam) = 0 THEN 'Geen' ELSE<br>to_char(count(sat.objectnaam), '999') END AS<br>"interessante satellieten"<br>FROM hemelobjecten planeet INNER JOIN hemelobjecten<br>ster ON (planeet.satellietvan = ster.objectnaam AND<br>ster.satellietvan is null)<br>LEFT OUTER JOIN hemelobjecten sat ON<br>(sat.satellietvan = planeet.objectnaam AND sat.diameter ><br>2000)<br>GROUP BY planeet.objectnaam<br>ORDER BY planeet.objectnaam     |  |
|  |  |  | SELECT planeet.objectnaam, CASE WHEN<br>count(sat.objectnaam) = 0 THEN 'Geen' ELSE<br>to_char(count(sat.objectnaam), '999') END AS "ver  |   |


















|  |  |   |  |   |
|--|--|---|--|---|
|  |  | <p>Geef een lijst van alle planeten die minstens 1 satelliet hebben op een afstand groter dan 500km. Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer omgekeerd alfabetisch op planeetnaam.</p>  | <p>verwijderde satellieten"<br/>FROM hemelobjecten planeet INNER JOIN hemelobjecten ster ON (planeet.satellietvan = ster.objectnaam AND ster.satellietvan is null)<br/>LEFT OUTER JOIN hemelobjecten sat ON (sat.satellietvan = planeet.objectnaam AND sat.afstand &gt; 500)<br/>GROUP BY planeet.objectnaam<br/>ORDER BY planeet.objectnaam desc</p>  |    |
|  |  | <p>Geef een lijst van alle planeten die minstens 1 satelliet hebben wiens naam een kleine letter i bevat. Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer op planeetnaam.</p>   | <p>SELECT planeet.objectnaam, CASE WHEN count(sat.objectnaam) = 0 THEN 'Geen' ELSE to_char(count(sat.objectnaam), '999') END AS "satellieten met een letter i"<br/>FROM hemelobjecten planeet INNER JOIN hemelobjecten ster ON (planeet.satellietvan = ster.objectnaam AND ster.satellietvan is null)<br/>LEFT OUTER JOIN hemelobjecten sat ON (sat.satellietvan = planeet.objectnaam AND sat.objectnaam LIKE "%i%")<br/>GROUP BY planeet.objectnaam<br/>ORDER BY planeet.objectnaam</p> |    |
|  |  | <p>Geef een overzicht van alle spelers bestaande uit het spelersnummer, de naam en hun geboortjaar</p>  | <p>SELECT spelersnr, naam, EXTRACT(YEAR FROM geb_datum) AS geboortjaar<br/>FROM spelers</p>  |    |
|  |  | <p>We willen van alle boetes de volgende feiten weten: gemiddelde, kleinste boete en grootste boete. Elk afgerond op 2 cijfers na de komma.</p>   | <p>SELECT round(avg(bedrag), 2) AS "gemiddelde boete", round(min(bedrag), 2) AS "kleinste boete", round(max(bedrag), 2) AS "grootste boete"<br/>FROM boetes</p>  |    |
|  |  | <p>We willen een vereenvoudigd, geïndexeerd overzicht van alles boetes.<br/>Boetes kleiner dan 25 worden weergegeven als "verwaarloosbaar" (tip: gebruik CASE), alle andere boetes moeten met 10% verhoogd worden. Geef voor elke boete ook de spelersnummer weer. Merk op dat je in 1 kolom bedragen als Strings moet voorstellen, gebruik hiervoor to_char, met een formatering van 3 getallen.</p> <p>Je hoeft niet te sorteren.</p> | <p>SELECT spelersnr, CASE WHEN bedrag &lt; 25 THEN 'verwaarloosbaar' ELSE to_char(bedrag*1.1, '999') END as bedrag<br/>FROM boetes</p>   |    |
|  |  | <p>Geef de totale som van het aantal betaalde boetes en daarnaast ook het totale aantal boetes.</p>   | <p>SELECT SUM(bedrag) as "som der boetes", count(*) as "aantal boetes"<br/>FROM boetes</p>   |    |
|  |  | <p>Selecteer de straatnamen die eindigen op weg.</p>  | <p>SELECT straat from spelers where straat like '%weg'</p>   |    |
|  |  | <p>Geef alle persoonnamen met een naam die meer dan 8 tekens telt</p>   | <p>SELECT naam as persoonnamen from spelers where length(naam) &gt; 8</p>  |   |
|  |  | <p>Tel hoeveel verschillende plaatsen er zijn.</p>  | <p>SELECT count(DISTINCT plaats) from spelers</p>  |  |
|  |  | <p>Selecteer de namen en geboortejaren van de leden. Geef de volledige naam weer in 1 kolom zodat je eerst de voorletters hebt, vervolgens de voorvoegsels en tot slot de naam.</p>   | <p>SELECT voorletters   ' '    naam as "oudste leden", EXTRACT(YEAR FROM geb_datum) as geboortjaar<br/>FROM spelers<br/>ORDER BY GEB_DATUM ASC;</p>  |  |
|  |  | <p>Zoek het gemiddelde toetredingsjaar (geen kommagetal toegelaten, rond af naar beneden). Merk op dat het type van je resultaat numeric moet zijn (en niet decimal bv.).</p>   | <p>SELECT FLOOR(AVG(jaartoe)) as "gemiddelde toetredingsjaar"<br/>from spelers</p>   |  |
|  |  | <p>Zoek de gemiddelde leeftijd in jaren in 2014 onafhankelijk van in welke maand de speler geboren is(geen kommagetal toegelaten, rond af naar boven)</p>   | <p>SELECT CEIL(AVG(2014 - EXTRACT(YEAR FROM GEB_DATUM))) as "gemiddelde leeftijd"<br/>from spelers</p>   |  |
|  |  | <p>Selecteer de naam van de spelers die Voorzitter geweest zijn, en de periode in 1 kolom. Sorteer van meest recent naar minst recent.</p>  | <p>SELECT s.naam as voorzitter, begin_datum    ' '    eind_datum as periode<br/>FROM SPELERS as s, BESTUURSLIEDEN as b<br/>WHERE s.spelersnr = b.spelersnr<br/>AND functie = 'Voorzitter'<br/>AND eind_datum IS NOT NULL<br/>ORDER BY periode desc</p>   |  |
|  |  | <p>Tel het aantal mensen die momenteel in het bestuur zitten, maar geen specifieke bestuursfunctie hebben (dus zij die momenteel enkel lid zijn).</p>   | <p>SELECT count(functie)<br/>from bestuursleden<br/>where functie = 'Lid' AND eind_datum IS NULL</p>   |  |
|  |  | <p>Geef per speler per wedstrijd dat hij gespeeld heeft, of hij gewonnen heeft met zijn team. Sorteer enkel op de naam van de speler (en niet op wedstrijdnummer).</p>  | <p>SELECT s.naam, w.wedstrijdnr, (CASE WHEN w.gewonnen &gt; verloren THEN 'gewonnen' ELSE 'verloren' END ) as "uitslag"<br/>from spelers as s, wedstrijden as w<br/>where s.spelersnr = w.spelersnr<br/>order by s.naam</p>  |  |
|  |  | <p>Genereer 'mevrouw &lt;naam&gt;' of 'meneer &lt;naam&gt;' naargelang het geslacht. Deze string moet in een kolom groot steken. Toon alle mannen bij elkaar, en vervolgens alle vrouwen. Gebruik niet de concat() functie.</p>   | <p>SELECT CASE WHEN s.geslacht = 'V' THEN 'mevrouw' ELSE 'meneer' END    ' '    s.naam as "groet"<br/>from spelers as s<br/>order by s.geslacht asc</p>  |  |
|  |  | <p>Selecteer de namen van voorzitters en de periode (= begindatum streepje einddatum) waarin ze voorzitter waren (of nog steeds zijn) en op welke datum ze beboet zijn geweest. Gebruik   .</p>   | <p>select s.naam, CASE WHEN be.eind_datum IS NOT NULL THEN to_char(be.begin_datum, 'DD/MM/YYYY')    ' - '    to_char(be.eind_datum, 'DD/MM/YYYY') ELSE to_char(be.begin_datum, 'DD/MM/YYYY')    ' - ' END as periode, bo.datum<br/>from bestuursleden as be, boetes as bo, spelers as s<br/>where be.spelersnr = bo.spelersnr<br/>and functie = 'Voorzitter'<br/>and s.spelersnr = be.spelersnr</p>  |  |
|  |  | <p>Geef de namen van alle spelers waarin de letter e tweemaal voorkomt, maar geen 3 keer. Geef van elke speler ook zijn leeftijd in jaren. Sorteer op geboortedatum zodat de oudste speler eerst wordt getoond. TIP: gebruik AGE().</p>   | <p>SELECT naam, extract(YEAR FROM age(geb_datum)) as leeftijd<br/>FROM spelers<br/>WHERE naam LIKE "%e%e%" AND NOT (naam LIKE "%e%e%e%")<br/>ORDER BY geb_datum ASC</p>  |  |












|  |  |   |  |   |
|--|--|---|--|---|
|  |  | Geef een lijst van alle spelers die momenteel een bestuursfunctie uitoefenen. Geef van deze mensen de naam en hun functie. Sorteert zodanig dat eerst die speler wordt getoond die reeds het langst zijn/haar bestuursfunctie uitoefent                           | SELECT naam, functie<br>FROM bestuursleden B, spelers S<br>WHERE eind_datum IS NULL AND B.spelersnr = S.spelersnr<br>ORDER BY begin_datum  |    |
|  |  | Geef een lijst van de spelernummers die momenteel nog bestuurslid zijn. Toon per spelersnummer ook de functie. Hint: EIND_DATUM IS NULL.  | select spelersnr, functie from bestuursleden<br>where eind_datum IS NULL;  |    |
|  |  | Geef een lijst van spelersnummers die nu geen bestuurslid meer zijn, maar die het wel ooit geweest zijn. Toon per spelersnummer zijn functie en de eind datum. Hint: EIND_DATUM NOT NULL  | select spelersnr, functie, eind_datum from bestuursleden<br>where eind_datum IS NOT NULL;  |    |
|  |  | Schrijf de query die het aantal rijen in boetes telt.   | select count(*) from boetes;   |    |
|  |  | Schrijf de query die het aantal gespeelde wedstrijden telt.   | select count(*) from wedstrijden;  |    |
|  |  | Schrijf de query die het aantal teams in de tweede divisie toont.   | select count(*) from teams where divisie = 'tweede';   |    |
|  |  | Geef een lijst van de spelers hun postcodes.  | select postcode from spelers;  |    |
|  |  | Geef de lijst van spelers die in een straat met naam laan of weg wonen.   | select naam, jaartoe from spelers<br>where straat LIKE '%weg' or straat LIKE '%laan';  |    |
|  |  | Schrijf de query die je het hoogst genoteerde boetebedrag geeft   | select max(bedrag) from boetes;  |    |
|  |  | Schrijf de query die je het jaar geeft waarop het eerste lid togetreden is.   | select min(jaartoe) from spelers;  |    |
|  |  | Schrijf de query die je de datum geeft waarop het eerste bestuurslid aan zijn functie begonnen is.  | select min(begin_datum) from bestuursleden;  |    |
|  |  | Geef een lijst van alle mogelijke dubbelcombinaties tussen spelers onderling. Sorteert op "eerste speler" en vervolgens op "tweede speler".   | select s.voorletters    ' '    s.naam as "eerste speler",<br>t.voorletters    ' '    t.naam as "tweede speler" from spelers<br>as s, spelers as t<br>where s.spelersnr <> t.spelersnr<br>order by "eerste speler", "tweede speler"   |    |
|  |  | Geef een lijst van alle mogelijke dubbelcombinaties van spelers. Beperking: de verschillende spelers mogen geen kapitein zijn van hetzelfde team (moest dit mogelijk zijn). Orden eerst naar speler1, dan naar speler2.   | select DISTINCT s.voorletters    ' '    s.naam AS "speler1",<br>t.voorletters    ' '    t.naam AS "speler2"<br>from spelers as s, spelers as t, teams as te, teams as tt<br>where s.spelersnr <> t.spelersnr AND<br>NOT (s.spelersnr = te.spelersnr AND t.spelersnr =<br>tt.spelersnr AND te.teamnr = tt.teamnr)<br>ORDER BY 1,2 |    |
|  |  | Geef voor alle effectieve ruimtebezoeken het bezochte hemelobject weer en de verblijfsduur. Sorteert aflopend op verblijfsduur  | SELECT objectnaam, verblijfsduur<br>FROM bezoeken<br>WHERE verblijfsduur > 0<br>ORDER BY 2 DESC  |  |
|  |  | Geef alle hemelobjecten weer met een diameter groter dan 50 en kleiner dan 10000. Sorteert eerst aflopend op diameter en dan alfabetisch op naam  | SELECT objectnaam, diameter<br>FROM hemelobjecten<br>WHERE diameter > 50 AND diameter < 10000<br>ORDER BY 2 DESC, 1 ASC  |  |
|  |  | Geef alle planeten weer met een diameter groter dan 10000. Sorteert alfabetisch op naam en dan op diameter. Een planeet is een hemelobject die een satelliet is van een hemelobject dat geen satelliet is. Hint: neem de tabel hemelobjecten tweemaal in je from. | SELECT p.objectnaam, p.diameter<br>FROM hemelobjecten p, hemelobjecten s<br>WHERE p.diameter > 10000 AND p.satellietvan =<br>s.objectnaam AND s.satellietvan is null<br>ORDER BY 1 ASC, 2 ASC  |  |
|  |  | Geef het geboortejaar van de jongste klant  | SELECT EXTRACT(YEAR FROM MAX(geboortedatum))<br>FROM klanten   |  |
|  |  | Geef de naam (= voornaam familienaam) van alle klanten die ooit al hebben deelgenomen aan een reis. Sorteert alfabetisch en zorg dat elke persoon slechts 1 maal voorkomt.  | SELECT DISTINCT K.vnaam    ' '    K.naam as naam<br>FROM klanten K, deelnames D<br>WHERE K.klantnr = D.klantnr<br>ORDER BY 1   |  |
|  |  | Geef voor klant met klantnr 121 het aantal reizen dat hij heeft gemaakt.  | SELECT count(reisnr) AS aantal_reizen<br>FROM deelnames<br>WHERE klantnr = 121   |  |
|  |  | Geef voor klant met klantnr 121 het aantal hemelobjecten dat hij effectief heeft bezocht over al zijn reizen heen. Hint: als hij de maan tweemaal bezocht heeft, tel je die dubbel.   | SELECT count(objectnaam) AS aantal_bezoeken<br>FROM deelnames D, bezoeken B<br>WHERE D.klantnr = 121 AND D.reisnr = B.reisnr   |  |
|  |  | Geef voor klant met klantnr 121 het aantal unieke hemelobjecten dat hij effectief heeft bezocht over al zijn reizen heen.   | SELECT count(DISTINCT objectnaam) AS<br>aantal_bezoeken<br>FROM deelnames D, bezoeken B<br>WHERE D.klantnr = 121 AND D.reisnr = B.reisnr   |  |
|  |  | Geef het bedrag van de duurste reis als "duurste"   | SELECT MAX(prijs) as duurste<br>FROM reizen  |  |
|  |  | Geef van alle reizen die langer dan 400 dagen duurden, een lijst van deelnemers. Output is reisnr, klantnr. Sorteert eerst op reisduur en dan op klantnr  | SELECT R.reisnr, K.klantnr<br>FROM reizen R, deelnames K<br>WHERE R.reisduur > 400 AND k.reisnr = R.reisnr<br>ORDER BY R.reisduur, 2   |  |
|  |  | Geef de totale verblijfsduur van alle bezoeken samen aan Mars   | SELECT sum(verblijfsduur)<br>FROM bezoeken<br>WHERE objectnaam = 'Mars'  |  |
|  |  | Geef het aantal satellieten van Jupiter   | SELECT count(objectnaam)<br>FROM hemelobjecten<br>WHERE satellietvan = 'Jupiter'   |  |
|  |  | Geef het aantal satellieten van Jupiter en Saturnus samen   | SELECT count(objectnaam)<br>FROM hemelobjecten<br>WHERE satellietvan IN ('Jupiter', 'Saturnus')  |  |

SELECT count(DISTINCT S.objectnaam) as

|      |                       |   |   |   |
|------|-----------------------|---|---|---|
|      |                       | Geef het aantal unieke satellieten van planeten binnen ons sterrenstelsel   | aantal_satellieten<br>FROM hemelobjecten Z, hemelobjecten P, hemelobjecten S<br>WHERE Z.satellietvan is null AND P.satellietvan = Z.objectnaam AND S.satellietvan = P.objectnaam  |     |
|      |                       | Geef de totale som van alle afstanden tot hun planeet van alle satellieten van Saturnus   | SELECT SUM(S.afstand) as "Totale afstand"<br>FROM hemelobjecten P, hemelobjecten S<br>WHERE P.objectnaam = 'Saturnus' AND S.satellietvan = P.objectnaam   |    |
|      |                       | Geef de gemiddelde afstanden tot hun planeet van alle satellieten van Saturnus, afgerond op 2 getallen na de komma  | SELECT round(AVG(S.afstand), 2) as "Gemiddelde afstand"<br>FROM hemelobjecten P, hemelobjecten S<br>WHERE P.objectnaam = 'Saturnus' AND S.satellietvan = P.objectnaam   |    |
| 2023 | Databanken:<br>Herh 1 | Geef voor elke wedstrijd het wedstrijdnummer en de volledige naam van de aanvoerder van het team waarvoor de wedstrijd werd gespeeld. Sorteer je resultaat volgens het wedstrijdnummer in oplopende volgorde.   | SELECT wedstrijdnr, naam, voorletters<br>FROM (wedstrijden W INNER JOIN teams T<br>ON W.teamnr = T.teamnr) INNER JOIN spelers S<br>ON T.spelersnr = S.spelersnr<br>ORDER BY 1   |    |
| 2023 | Databanken:<br>Herh 1 | Geef voor alle huidige bestuurleden hun functie en de lijst van boetes die voor hen werd betaald. Omdat je dit wil vergelijken met de boetebedragen die betaald werden voor leden die niet in het bestuur zitten, wil je deze boetebedragen ook opnemen in de tweede kolom van je resultaat. Sorteer je antwoord eerst op functie en daarna op het boetebedrag. | SELECT functie, bedrag<br>FROM boetes BT FULL OUTER JOIN bestuursleden B<br>ON BT.spelersnr = B.spelersnr<br>WHERE eind_datum is null<br>ORDER BY 1,2;  |    |
| 2023 | Databanken:<br>Herh 1 | Geef voor elke mannelijke speler wiens naam minstens 2 keer de letter 'e' bevat zijn functie die hij op dit moment uitoefent, als die er op dit moment één heeft. Sorteer op naam en functie.   | select naam, geslacht, functie<br>from spelers left outer join bestuursleden<br>on spelers.spelersnr = bestuursleden.spelersnr<br>and bestuursleden.eind_datum is null<br>where geslacht = 'M'<br>and naam like '%e%e%'<br>ORDER BY naam, functie;  |    |
| 2023 | Databanken:<br>Herh 1 | Geef alle spelers die geen boete gekregen hebben en niet in Den Haag wonen. Sorteer op jaar van toetreden nr en verder op de volgende kolommen.   | SELECT spelers.spelersnr, naam, plaats<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr = boetes.spelersnr<br>WHERE boetes.betalingsnr IS NULL<br>AND spelers.plaats <> 'Den Haag'<br>ORDER BY jaartoe, 1, 2 ,3;  |    |
| 2023 | Databanken:<br>Herh 1 | Geef per team de verloren wedstrijden. Zorg dat teams zonder verloren wedstrijden ook in de output verschijnen. Duid per wedstrijd aan of het om een actief bestuurslid gaat. Sorteer op divisie en wedstrijdnummer.  | SELECT teams.teamnr, teams.divisie,<br>wedstrijden.wedstrijdnr, wedstrijden.spelersnr,<br>CASE<br>WHEN bestuursleden.spelersnr IS NULL THEN 'I'<br>ELSE 'actief'<br>END AS bestuurslid<br>FROM teams<br>LEFT OUTER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr AND wedstrijden.verloren > wedstrijden.gewonnen<br>LEFT OUTER JOIN bestuursleden ON wedstrijden.spelersnr = bestuursleden.spelersnr AND bestuursleden.eind_datum IS NULL<br>ORDER BY divisie, wedstrijden.wedstrijdnr; |  |
| 2023 | Databanken:<br>Herh 1 | Geef een alfabetisch gesorteerde lijst van de namen van alle leden van de tennisvereniging die nog geen wedstrijden gespeeld hebben   | SELECT naam<br>FROM spelers S left outer join wedstrijden W<br>ON S.spelersnr = W.spelersnr<br>WHERE W.wedstrijdnr is null<br>ORDER BY naam;  |  |
| 2023 | Databanken:<br>Herh 1 | Geef een aflopend gesorteerde lijst van de nummers van alle spelers waarvoor nog geen boete werd betaald en die nog nooit in het bestuur van de tennisvereniging hebben gezeten.  | SELECT S.spelersnr<br>FROM spelers S LEFT OUTER JOIN boetes BT<br>ON S.spelersnr = BT.spelersnr<br>LEFT OUTER JOIN bestuursleden B<br>ON S.spelersnr = B.spelersnr<br>WHERE BT.spelersnr is null<br>AND B.spelersnr is null<br>ORDER BY 1 DESC;   |  |
| 2023 | Databanken:<br>Herh 1 | Geef de nummers van alle spelers (ook al hebben ze geen wedstrijd gespeeld), een lijst van de nummers van alle wedstrijden die ze gewonnen hebben, alsook het verschil in sets waarmee ze gewonnen hebben. Toon je resultaten gesorteerd volgens dit verschil van groot naar klein en oplopend op spelersnr.  | SELECT S.spelersnr, wedstrijdnr, gewonnen - verloren as verschil<br>FROM spelers S LEFT OUTER JOIN wedstrijden W<br>ON S.spelersnr = W.spelersnr AND gewonnen > verloren<br>ORDER BY verschil DESC, S.spelersnr;  |  |
| 2023 | Databanken:<br>Herh 1 | Geef van elke speler (die wedstrijden gespeeld heeft en boetes gekregen heeft) wonend in Rijswijk het spelersnr, de naam, de lijst met boetebedragen en de lijst met teams waarvoor hij/zij een wedstrijd gespeeld heeft. Geef het resultaat volgens oplopend spelersnr en boetebedrag.   | SELECT s.spelersnr, s.naam, bedrag, teamnr<br>FROM ((spelers s INNER JOIN boetes b ON s.spelersnr = b.spelersnr)<br>INNER JOIN wedstrijden w ON w.spelersnr = s.spelersnr)<br>WHERE plaats = 'Rijswijk'<br>ORDER BY s.spelersnr, b.bedrag   |  |
| 2023 | Databanken:<br>Herh 1 | Geef het spelersnummer en bondsnummer van alle spelers die jonger zijn dan de speler met bondsnummer 8467. gebruik een INNER JOIN. Sorteer 1,2  | SELECT S.spelersnr, S.bondsnr<br>FROM spelers as S INNER JOIN spelers as P<br>ON S.geb_datum > P.geb_datum<br>WHERE P.bondsnr = '8467'<br>ORDER BY 1, 2;  |  |
| 2023 | Databanken:<br>Herh 1 | Geef een lijst met het spelersnummer, de naam van de speler, de datum van de boete en het bedrag van de boete van al de spelers die een boete gekregen hebben met een bedrag groter dan 45,50 euro en in Rijswijk wonen. Geef expliciet aan welke join je gebruikt.   | SELECT spelers.spelersnr, spelers.naam, boetes.datum,<br>boetes.bedrag<br>FROM spelers<br>INNER JOIN boetes ON spelers.spelersnr = boetes.spelersnr<br>WHERE boetes.bedrag > 45.50<br>AND spelers.plaats = 'Rijswijk';  |  |
| 2023 | Databanken:           | Maak een lijst van alle mannelijke aanvoerders van een team en hun gespeelde wedstrijden. Hierbij toon je voor deze spelers het spelersnummer en de   | SELECT spelers.spelersnr, spelers.naam,<br>spelers.voorletters, teams.divisie, wedstrijden.wedstrijdnr<br>FROM spelers<br>INNER JOIN teams ON spelers.spelersnr = teams.spelersnr   |  |











|      |                    |   |   |   |
|------|--------------------|---|---|---|
|      | Herh 1             | volledige naam, voor het team de divisie en voor de wedstrijd het wedstrijdnummer.<br>Sorteer ook aflopend op het wedstrijdnummer.  | LEFT OUTER JOIN wedstrijden ON spelers.spelersnr = wedstrijden.spelersnr<br>WHERE spelers.geslacht = 'M'<br>ORDER BY wedstrijden.wedstrijdnr DESC;  |   |
| 2023 | Databanken: Herh 1 | Geef voor alle vrouwelijke spelers die in Den Haag, Zoetermeer of Leiden wonen het spelersnummer, hun woonplaats en een lijst van de teams waarvoor ze ooit gespeeld hebben, als ze ooit een wedstrijd gespeeld hebben. sorteer volgens 1,2,3   | SELECT S.spelersnr, plaats, teamnr<br>FROM spelers S LEFT OUTER JOIN wedstrijden W ON S.spelersnr = W.spelersnr<br>WHERE geslacht = 'V'<br>AND plaats in ('Den Haag', 'Zoetermeer', 'Leiden')<br>ORDER BY 1,2,3   |    |
| 2023 | Databanken: Herh 1 | Geef het gemiddeld boetebedrag per speler, afgerond op twee cijfers na de komma. spelers zonder boete moeten bovenaan staan met 'geen boetes'<br>Sorteer daarna op spelersnaam en vervolgens op gemiddeld boetebedrag.<br>Gebruik als datatype van de 2de kolom varchar(8) voor spelers die wel boetes hebben.                                      | SELECT spelers.naam,<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 'geen boetes'<br>ELSE CAST(ROUND(AVG(boetes.bedrag), 2) AS varchar(8))<br>END<br>AS gemiddeld<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr = boetes.spelersnr<br>GROUP BY spelers.spelersnr, spelers.naam<br>ORDER BY<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 0<br>ELSE 1<br>END, spelers.naam, AVG(boetes.bedrag);  |    |
| 2023 | Databanken: Herh 1 | Geef voor de actieve bestuursleden zonder boete hun laatste gespeelde wedstrijd (die met het hoogste wedstrijdnummer). Sorteer aflopend op spelersnr.   | SELECT bestuursleden.spelersnr,<br>MAX(wedstrijden.wedstrijdnr) AS laatstewedstrijd<br>FROM bestuursleden<br>INNER JOIN wedstrijden ON bestuursleden.spelersnr = wedstrijden.spelersnr AND bestuursleden.eind_datum IS NULL<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr = boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY bestuursleden.spelersnr<br>ORDER BY spelersnr DESC;   |    |
| 2023 | Databanken: Herh 1 | Geef per team de leeftijd van de aanvoerder (tip: postgresql heeft een AGE() functie) en het aantal verschillende spelers dat voor dit team gespeeld heeft. Alleen teams waarvoor wedstrijden zijn gespeeld en die een aanvoerder hebben, moeten vermeld worden. Sorteer op leeftijd en daarna op aantal verschillende spelers en daarna op teamnr. | SELECT teams.teamnr, CAST(EXTRACT(year from AGE(NOW(), spelers.geb_datum)) AS varchar(2))    ' jaar'<br>AS leeftijd, COUNT(DISTINCT wedstrijden.spelersnr) AS aantalspelers<br>FROM teams<br>INNER JOIN spelers ON teams.spelersnr = spelers.spelersnr<br>INNER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr<br>GROUP BY teams.teamnr, spelers.geb_datum<br>ORDER BY leeftijd, aantalspelers, teamnr;  |    |
| 2023 | Databanken: Herh 1 | Geef per team het hoogste wedstrijdnummer van een wedstrijd, gespeeld door een bestuurslid (actief en niet meer actief) die geen boete heeft gekregen. Sorteer op teamnr.   | SELECT teams.teamnr, MAX(wedstrijden.wedstrijdnr) AS laatstewedstrijd<br>FROM teams<br>INNER JOIN wedstrijden on teams.teamnr = wedstrijden.teamnr<br>INNER JOIN bestuursleden ON wedstrijden.spelersnr = bestuursleden.spelersnr<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr = boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY teams.teamnr<br>ORDER BY teams.teamnr;  |  |
| 2023 | Databanken: Herh 1 | Geef alle spelers die meer wedstrijden gespeeld hebben dan het aantal wedstrijden dat de huidige voorzitter heeft verloren. De huidige voorzitter komt zelf niet in de lijst voor. Gebruik geen subqueries. Sorteer op spelersnr.   | SELECT spelers.spelersnr<br>FROM spelers<br>INNER JOIN wedstrijden ON spelers.spelersnr = wedstrijden.spelersnr,<br>bestuursleden<br>INNER JOIN wedstrijden voorzitterswedstrijden ON bestuursleden.spelersnr = voorzitterswedstrijden.spelersnr<br>AND voorzitterswedstrijden.gewonnen < voorzitterswedstrijden.verloren<br>WHERE bestuursleden.eind_datum IS NULL AND bestuursleden.functie = 'Voorzitter' AND bestuursleden.spelersnr <> spelers.spelersnr<br>GROUP BY spelers.spelersnr<br>HAVING COUNT(DISTINCT wedstrijden.wedstrijdnr) > COUNT(DISTINCT voorzitterswedstrijden.wedstrijdnr)<br>ORDER BY spelers.spelersnr; |  |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Genereer startende van een integer array de volgende output: [[1,5],[99,100]]   | SELECT array_to_json('{{1,5},{99,100}}'::int[]);  |  |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Genereer startende van een heterogene array de volgende output: [1, 2, "3", 4, 5]   | SELECT json_build_array(1,2,'3',4,5)  |  |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Selecteer in onderstaande json string '{"a":[1,2,3],"b":[4,5,6]}' het tweede object   | SELECT '{"a":[1,2,3],"b":[4,5,6]}':json -> 'b'  |  |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Expandeer het buitenste JSON object uit de string '{"a":"foo", "b":"bar"}'.   | select * from json_each('{{"a":"foo", "b":"bar"}}')   |  |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Selecteer in onderstaande json string '{"1":["1,2,3"],"2":["4,5,6]}' het tweede object  | select '{"1":["1,2,3"],"2":["4,5,6]}':json->'2'   |  |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Selecteer in onderstaande json string '{"1":["1,2,3","2":["4,5,6]}' het eerste object   | select '{"1":["1,2,3"],"2":["4,5,6]}':json->'1'   |  |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Selecteer in onderstaande json string '{"1":["2"],"4":["5]}' het eerste object  | select '{"1":["2"],"4":["5]}':json->0   |  |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Selecteer in onderstaande json string '{"1":["2"],"4":["5]}' het tweede object  | select '{"1":["2"],"4":["5]}':json->1   |  |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Selecteer in onderstaande json string '{"1":["2"],"4":["5"],"6":[]}'  | select '{"1":["2"],"4":["5"],"6":[]}:json->2  |  |



















|      |                    |  |  |   |
|------|--------------------|--|--|---|
|      | JSON               | '[{"1":"2"}, {"4":"5"}, "6"]' het derde object   |  |   |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Selecteer uit onderstaande json string '{"a": {"b":{"c": "foo"}, "c":{"5":"6"}}}' het element "6"  | SELECT '{"a": {"b":{"c": "foo"}, "c":{"5":"6"}}}':json#>'{a,c,5}'  |    |
| 2023 | Databanken: JSON   | Gebruik JSON instructies. Selecteer uit onderstaande json string '{"a": {"b":{"c": "foo"}, "c":{"5":"6"}}}' het element "{c: "foo"}"   | SELECT '{"a": {"b":{"c": "foo"}, "c":{"5":"6"}}}':json#>'{a,b}'  |    |
| 2023 | Databanken: Herh 2 | Welke reizigers hebben al meer dan 1 reis ondernomen waarvoor ze meer dan 2,5 miljoen euro moesten betalen?<br>Sorteer op naam   | SELECT klanten.naam, COUNT(*) AS aantal_reizen<br>FROM reizen<br>INNER JOIN deelnames ON reizen.reisnr = deelnames.reisnr<br>INNER JOIN klanten ON deelnames.klantnr = klanten.klantnr<br>WHERE reizen.prijs > 2.50<br>GROUP BY klanten.klantnr, klanten.naam<br>HAVING COUNT(*) > 1<br>ORDER BY naam;   |    |
| 2023 | Databanken: Herh 2 | Op welke planeten verblijft men gemiddeld langer dan 2 dagen?<br>Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam, AVG(verblijfsduur)<br>FROM bezoeken<br>INNER JOIN hemelobjecten ON bezoeken.objectnaam = hemelobjecten.objectnaam<br>WHERE hemelobjecten.satellietvan = 'Zon'<br>GROUP BY hemelobjecten.objectnaam<br>HAVING AVG(verblijfsduur) > 2<br>ORDER BY hemelobjecten.objectnaam;   |    |
| 2023 | Databanken: Herh 2 | Bereken voor de klant wiens naam begint met G en eindigt met s hoeveel hij in totaal al besteed heeft aan reizen.  | SELECT klanten.naam, SUM(prijs)<br>FROM reizen<br>INNER JOIN deelnames ON reizen.reisnr = deelnames.reisnr<br>INNER JOIN klanten ON deelnames.klantnr = klanten.klantnr<br>WHERE klanten.naam LIKE 'G%s'<br>GROUP BY klanten.naam;   |    |
| 2023 | Databanken: Herh 2 | Maak een lijst met een overzicht van de reizen met deelnemers. Geef per reis het aantal deelnemers van elke reis. Orden de lijst dalend op basis van het aantal deelnemers, als er eenzelfde aantal deelnemers is, moeten deze stijgend geordend worden volgens reisnummer.                                      | SELECT reizen.reisnr, COUNT(*) AS deelnemers<br>FROM reizen<br>INNER JOIN deelnames ON reizen.reisnr = deelnames.reisnr<br>GROUP BY reizen.reisnr<br>ORDER BY deelnemers DESC, reisnr;   |    |
| 2023 | Databanken: Herh 2 | Geef een lijst van alle reizen met tenminste 1 bezoek of passage aan de Maan of aan Mars. De lijst moet stijgend gesorteerd zijn op basis van het vertrekdatum.  | SELECT DISTINCT bezoeken.reisnr, reizen.vertrekdatum<br>FROM bezoeken<br>INNER JOIN reizen ON bezoeken.reisnr = reizen.reisnr<br>WHERE objectnaam IN ('Maan', 'Mars')<br>ORDER BY vertrekdatum;  |    |
| 2023 | Databanken: Herh 2 | Maak een lijst met een overzicht van de reizen en het aantal deelnemers van elke reis. Orden de lijst dalend op basis van het aantal deelnemers, als er eenzelfde aantal deelnemers is, moeten deze stijgend geordend worden volgens reisnummer.<br>Zorg dat reizen zonder deelnames ook in het resultaat staan. | SELECT reizen.reisnr, COUNT(deelnames.reisnr) AS deelnemers<br>FROM reizen<br>LEFT OUTER JOIN deelnames ON reizen.reisnr = deelnames.reisnr<br>GROUP BY reizen.reisnr<br>ORDER BY deelnemers DESC, reisnr;   |  |
| 2023 | Databanken: Herh 2 | Welke reizen hebben exact drie verschillende hemelobjecten als reisdoel?<br>Sorteer op reisnr.   | SELECT reisnr<br>FROM bezoeken<br>GROUP BY reisnr<br>HAVING COUNT(DISTINCT objectnaam) = 3<br>ORDER BY reisnr;   |  |
| 2023 | Databanken: Herh 2 | Welke klanten (klantnaam, geboortedatum) zijn op een ruimtereis vertrokken in het jaar dat ze 45 jaar geworden zijn?<br>Het resultaat moet stijgend gesorteerd worden op de geboortedatum.   | SELECT DISTINCT naam AS klantnaam, klanten.geboortedatum<br>FROM klanten<br>INNER JOIN deelnames ON klanten.klantnr = deelnames.klantnr<br>INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br>WHERE EXTRACT(YEAR FROM reizen.vertrekdatum) - EXTRACT(YEAR FROM klanten.geboortedatum) = 45<br>ORDER BY klanten.geboortedatum;  |  |
| 2023 | Databanken: Herh 2 | Welke planeten met meer dan 7 manen worden bezocht (met of zonder verblijf)?<br>Sorteer aflopend op basis van het aantal manen.<br>Let erop dat je planeten die meerdere keren bezocht worden, niet dubbel telt.   | SELECT planeten.objectnaam<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten satellieten ON satellieten.satellietvan = planeten.objectnaam<br>INNER JOIN bezoeken ON planeten.objectnaam = bezoeken.objectnaam<br>WHERE planeten.satellietvan = 'Zon'<br>GROUP BY planeten.objectnaam<br>HAVING COUNT(DISTINCT satellieten.objectnaam) > 7<br>ORDER BY COUNT(DISTINCT satellieten.objectnaam) DESC; |  |
| 2023 | Databanken: Herh 2 | Bereken voor alle hemelobjecten die satellieten hebben, het aantal satellieten per hemelobject.<br>De lijst moet dalend gesorteerd worden op basis van het aantal satellieten van de hemelobjecten en daarna alfabetisch op basis van objectnaam.  | SELECT planeten.objectnaam, COUNT(*)<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten satellieten ON satellieten.satellietvan = planeten.objectnaam<br>GROUP BY planeten.objectnaam<br>ORDER BY COUNT(*) DESC, planeten.objectnaam;  |  |
| 2023 | Databanken: Herh 2 | Geef een lijst met alle planeten en per planeet zijn satellieten.<br>Sorteer op planeet en daarna op satelliet.  | SELECT planeten.objectnaam as planeet, satellieten.objectnaam AS maan<br>FROM hemelobjecten planeten<br>LEFT OUTER JOIN hemelobjecten satellieten ON satellieten.satellietvan = planeten.objectnaam<br>WHERE planeten.satellietvan = 'Zon'<br>ORDER BY planeet, maan;  |  |
| 2023 | Databanken: Herh 2 | Geef de leeftijd van de jongste klant op moment van vertrek.   | SELECT EXTRACT (YEAR FROM MIN(AGE(vertrekdatum, geboortedatum))) AS jongsteleeftijd<br>FROM klanten<br>INNER JOIN deelnames ON klanten.klantnr = deelnames.klantnr<br>INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr;   |  |











|      |                       |  |   |   |
|------|-----------------------|--|---|---|
| 2023 | Databanken:<br>Herh 2 | Geef de diameter van de grootste, niet bezochte maan (satelliet van een planeet).  | SELECT MAX(manen.diameter) AS grootstemaan<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten manen ON<br>planeten.objectnaam = manen.satellietvan AND<br>planeten.satellietvan = 'Zon'<br>LEFT OUTER JOIN bezoeken ON manen.objectnaam =<br>bezoeken.objectnaam<br>WHERE bezoeken.reisnr IS NULL;  |    |
| 2023 | Databanken:<br>Herh 2 | Geef de naam, diameter en het langste verblijf op van alle hemelobjecten met minder dan 5 hemelobjecten die rond dit object draaien.<br><br>Anders gezegd: geef van alle bezochte hemelobjecten de naam, diameter en het langste verblijf; als deze bezochte hemelobjecten minder dan 5 satellieten hebben.<br><br>Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam,<br>hemelobjecten.diameter, MAX(bezoeken.verblijfsduur) AS<br>maximale_verblijf<br>FROM bezoeken<br>INNER JOIN hemelobjecten ON bezoeken.objectnaam =<br>hemelobjecten.objectnaam<br>LEFT OUTER JOIN hemelobjecten manen ON<br>hemelobjecten.objectnaam = manen.satellietvan<br>GROUP BY hemelobjecten.objectnaam,<br>hemelobjecten.diameter<br>HAVING COUNT(DISTINCT manen.objectnaam) < 5<br>ORDER BY objectnaam; |    |
| 2023 | Databanken:<br>Herh 2 | Geef per klant het totaal aantal reizen waaraan deze klant zal deelnemen en het langste bezoek dat deze klant zal maken aan een hemelobject, over alle reizen heen.<br>Klanten zonder reizen of zonder bezoeken moeten ook voorkomen in het overzicht.<br>Sorteer op klantnr.  | SELECT klanten.klantnr, COUNT(DISTINCT reizen.reisnr)<br>AS aantal, MAX(bezoeken.verblijfsduur) AS langstebezoek<br>FROM klanten<br>LEFT OUTER JOIN deelnames ON klanten.klantnr =<br>deelnames.klantnr<br>LEFT OUTER JOIN reizen ON deelnames.reisnr =<br>reizen.reisnr<br>LEFT OUTER JOIN bezoeken ON deelnames.reisnr =<br>bezoeken.reisnr<br>GROUP BY klanten.klantnr<br>ORDER BY klanten.klantnr;  |    |
| 2023 | Databanken:<br>Herh 2 | Geef voor elk hemelobject de minimale en maximale gemiddelde afstand tot zijn zon (de centrale ster in een sterrenstelsel) als je weet dat de kolom 'afstand' de gemiddelde afstand bevat tot het hemelobject waar rond ze draaien. Met de grootte van het hemelobject hoef je geen rekening te houden.<br>Sorteer op minimale afstand en op objectnaam.<br>Tip: bekijk de functie COALESCE om met null-waardes rekening te houden.<br>Opm: berekende afstanden dienen altijd positief te zijn, je kan eventueel abs() functie gebruiken | SELECT<br>hemelobjecten.objectnaam,<br>coalesce(hemelobjecten.afstand, 0) +<br>coalesce(draaitrond.afstand, 0) AS maximale_afstand,<br>abs(coalesce(draaitrond.afstand, 0) -<br>coalesce(hemelobjecten.afstand, 0)) AS minimale_afstand<br>FROM hemelobjecten<br>LEFT OUTER JOIN hemelobjecten draaitrond ON<br>draaitrond.objectnaam = hemelobjecten.satellietvan<br>ORDER BY minimale_afstand, objectnaam;  |    |
| 2023 | Databanken:<br>Subq 1 | Geef voor elke speler die een wedstrijd heeft gespeeld het spelersnr en het totaal aantal boetes. Spelers die een wedstrijd gespeeld hebben, maar geen boetes hebben, moeten ook getoond worden.<br>Sorteer op het aantal boetes en op spelersnr;  | SELECT DISTINCT w.spelersnr, aantalboetes<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes<br>RIGHT OUTER JOIN wedstrijden w USING (spelersnr)<br>ORDER BY aantalboetes   |    |
| 2023 | Databanken:<br>Subq 1 | Geef voor alle spelers die geen penningmeester zijn of zijn geweest alle gewonnen wedstrijden, gesorteerd op wedstrijdnummer.  | SELECT spelersnr, wedstrijdnr<br>FROM wedstrijden<br>WHERE spelersnr NOT IN (<br>SELECT spelersnr<br>FROM bestuursleden<br>WHERE functie = 'Penningmeester')<br>AND gewonnen > verloren<br>ORDER BY wedstrijdnr;  |  |
| 2023 | Databanken:<br>Subq 1 | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer op spelersnr. Toon 3 getallen na de komma, zet het verschil om naar het numeric type met precisie van 5 en een schaal van 3.   | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe-<br>avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1   |  |
| 2023 | Databanken:<br>Subq 1 | Je kan per speler berekenen hoeveel boetes die speler heeft gehad en wat het totaalbedrag per speler is. Pas nu deze query aan zodat per verschillend aantal boetes wordt getoond hoe vaak dit aantal boetes voorkwam.Sorteer eerst op de eerste kolom en daarna op de tweede kolom.   | SELECT a, count(a)<br>FROM (SELECT count(bedrag) AS a<br>FROM boetes<br>GROUP BY spelersnr) b<br>GROUP BY a<br>ORDER BY 1,2   |  |
| 2023 | Databanken:<br>Subq 1 | Geef van alle spelers het verschil tussen het jaar van toetreding en het geboortjaar, maar geef alleen die spelers waarvan dat verschil groter is dan 20. Sorteer deze gegevens beginnend bij de eerste kolom, eindigend bij de laatste kolom.   | SELECT s.spelersnr, s.naam, s.voorletters, (s.jaartoe -<br>extract(year from s.geb_datum)) AS Toetredingsleeftijd<br>FROM spelers s<br>WHERE (s.jaartoe - extract(year from s.geb_datum)) > 20<br>ORDER BY 1,2,3,4  |  |
| 2023 | Databanken:<br>Subq 1 | Geef voor elke aanvoerder het spelersnr, de naam en het aantal boetes dat voor hem of haar betaald is en het aantal teams dat hij of zij aanvoert. Aanvoerders zonder boetes mogen niet getoond worden. Sorteer, beginnend bij de eerste kolom, eindigend bij de laatste kolom.  | SELECT s.spelersnr, s.naam, s.voorletters, aantalboetes,<br>aantalteams<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes,<br>(SELECT spelersnr, count(*) AS aantalteams<br>FROM teams<br>GROUP BY spelersnr) AS aantal_teams,<br>spelers s<br>WHERE s.spelersnr=aantal_boetes.spelersnr<br>AND s.spelersnr=aantal_teams.spelersnr<br>ORDER BY 1,2,3,4,5                                   |  |
| 2023 | Databanken:<br>Subq 1 | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer op spelersnr. Zet het berekende verschil om naar het datatype numeric met precisie 5 en schaal 3.  | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe-<br>avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1   |  |
|      |                       | Geef een lijst van alle spelers (spelersnr en woonplaats) die met  | SELECT spelersnr, plaats<br>FROM spelers<br>WHERE plaats IN (<br>SELECT plaats  |   |

|      |                                     |   |   |   |
|------|-------------------------------------|---|---|---|
| 2023 | Databanken:<br>Subq 1               | minstens twee in dezelfde plaats wonen. Sorteer aflopend op woonplaats, daarna op spelersnr.  | FROM spelers<br>GROUP BY plaats<br>HAVING COUNT(*) >= 2<br>)<br>ORDER BY plaats DESC, spelersnr;  |     |
| 2023 | Databanken:<br>Subq 1               | Geef de spelersgegevens van de speler(s) met het hoogste bedrag (voor één boete, niet het totaalbedrag). Als twee spelers een even hoge boete gehad hebben, moeten beide spelers getoond worden (LIMIT is dus geen optie).<br>Sorteer alfabetisch op naam en voorletters.   | SELECT spelers.spelersnr, voorletters, naam<br>FROM spelers<br>INNER JOIN boetes using (spelersnr)<br>WHERE boetes.bedrag = (<br>SELECT MAX(bedrag)<br>FROM boetes<br>)<br>ORDER BY naam, voorletters   |    |
| 2023 | Databanken:<br>Subq 1               | We willen een statistiek van hoeveel wedstrijden de spelers winnen.<br>Geef een lijst met aantal gewonnen wedstrijden en het aantal spelers dat dit aantal wedstrijden gewonnen heeft.<br>Dus bv als er vier spelers zijn die elk drie wedstrijden hebben gewonnen, dan is de output: aantal_gewonnen: 3, aantal_spelers: 4. Dit graag voor alle aantallen gewonnen wedstrijden en alle spelers.<br>Sorteer op aantal gewonnen wedstrijden. | SELECT aantal_gewonnen, count(spelersnr) AS<br>aantal_spelers<br>FROM (<br>SELECT spelersnr, count(*) AS aantal_gewonnen<br>FROM wedstrijden<br>WHERE gewonnen > verloren<br>GROUP BY spelersnr) AS gewonnen<br>GROUP BY aantal_gewonnen<br>ORDER BY aantal_gewonnen;   |    |
| 2023 | Databanken:<br>Subq 1<br>verbreding | Geef een lijst van alle hemelobjecten die meer keer bezocht gaan worden dan Jupiter. (onafhankelijk van het aantal deelnames)<br>Sorteer op objectnaam, diameter.   | SELECT hemelobjecten.objectnaam,<br>hemelobjecten.diameter<br>FROM hemelobjecten<br>INNER JOIN bezoeken USING (objectnaam)<br>GROUP BY hemelobjecten.objectnaam,<br>hemelobjecten.diameter<br>HAVING COUNT(*) > (<br>SELECT COUNT(*)<br>FROM bezoeken<br>WHERE objectnaam = 'Jupiter'<br>)<br>ORDER BY hemelobjecten.objectnaam,<br>hemelobjecten.diameter  |    |
| 2023 | Databanken:<br>Subq 1<br>verbreding | Geef het hemellichaam dat het laatst bezocht is.<br>Gebruik hiervoor de laatste vertrekdatum van de reis en laatste volgnummer van bezoek. Tip: gebruik hiervoor een rij-subquery.<br>Gebruik geen limit of top.  | SELECT objectnaam<br>FROM bezoeken<br>WHERE (reisnr, volgnr) = (<br>SELECT bezoeken.reisnr, MAX(bezoeken.volgnr)<br>FROM bezoeken<br>INNER JOIN reizen laatstereis USING (reisnr)<br>WHERE laatstereis.vertrekdatum = (<br>SELECT MAX(vertrekdatum)<br>FROM reizen<br>)<br>GROUP BY bezoeken.reisnr<br>)  |    |
| 2023 | Databanken:<br>Subq 1<br>verbreding | Geef het reisnr, de prijs en vertrekdatum van de reis met de hoogste gemiddelde verblijfsduur op een hemelobject (=som van de verblijfsduur / aantal bezoeken per reis).  | SELECT reizen.reisnr, prijs, vertrekdatum<br>FROM reizen<br>INNER JOIN bezoeken USING (reisnr)<br>GROUP BY reizen.reisnr, prijs, vertrekdatum<br>HAVING (SUM(verblijfsduur) / COUNT(objectnaam)) = (<br>SELECT MAX(gemiddelde)<br>FROM (<br>SELECT SUM(verblijfsduur) / COUNT(objectnaam) as<br>gemiddelde<br>FROM bezoeken<br>GROUP BY reisnr<br>) AS reisgemiddelden<br>)   |  |
| 2023 | Databanken:<br>Subq 1<br>verbreding | Geef de planeet (draait dus rond de zon) met de meeste satellieten.<br>Sorteer op objectnaam.   | SELECT planeten.objectnaam<br>FROM hemelobjecten planeten<br>LEFT OUTER JOIN hemelobjecten satellieten ON<br>planeten.objectnaam = satellieten.satellietvan<br>WHERE planeten.satellietvan = 'Zon'<br>GROUP BY planeten.objectnaam<br>HAVING COUNT(*) = (<br>SELECT MAX(aantalsatellieten)<br>FROM (<br>SELECT COUNT(*) AS aantalsatellieten<br>FROM hemelobjecten planeten<br>LEFT OUTER JOIN hemelobjecten satellieten ON<br>planeten.objectnaam = satellieten.satellietvan<br>WHERE planeten.satellietvan = 'Zon'<br>GROUP BY planeten.objectnaam<br>) AS aantallen<br>)<br>ORDER BY planeten.objectnaam |  |
| 2023 | Databanken:<br>Subq 1<br>verbreding | Geef het op één na kleinste hemellichaam.<br>Je kan dit vinden door handig gebruik te maken van expliciete joins en een doorsnedevoorwaarde. Tip: probeer eerst een lijst te krijgen van alle hemelobjecten en het aantal hemellichaam dat kleiner is dan dat hemelobject.  | SELECT zelf.objectnaam, count(kleiner.objectnaam) AS<br>aantalkleiner<br>FROM hemelobjecten zelf<br>LEFT OUTER JOIN hemelobjecten kleiner ON<br>zelf.diameter > kleiner.diameter<br>GROUP BY zelf.objectnaam<br>HAVING count(kleiner.objectnaam) = 1  |  |
| 2023 | Databanken:<br>Subq 2               | Geef het totaal aantal boetes, het totale boetebedrag, het minimum en het maximum boetebedrag dat door onze club betaald werd. Let er hierbij op dat er gehele getallen worden getoond (rond af indien nodig). Sorteer van voor naar achter, oplopend. Deze opgave behoeft geen subquery.   | SELECT count(*) AS Aantal_boetes, round(sum(bedrag))<br>AS Totaal_bedrag, round(min(bedrag)) as Minimum,<br>round(max(bedrag)) as Maximum<br>FROM boetes<br>ORDER BY 1,2,3,4;   |  |
| 2023 | Databanken:<br>Subq 2               | Geef voor elke aanvoerder het spelersnr, de naam en het aantal boetes dat voor hem of haar betaald is en het aantal teams dat hij of zij aanvoert. Toon enkel aanvoerders die boetes gekregen hebben. Sorteer van voor naar achter, oplopend.   | SELECT s.spelersnr, s.naam, s.voorletters, aantalboetes,<br>aantalteams<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes,<br>(SELECT spelersnr, count(*) AS aantalteams<br>FROM teams<br>GROUP BY spelersnr) AS aantal_teams,<br>spelers s  |  |

|      |                                     |   |   |   |
|------|-------------------------------------|---|---|---|
|      |                                     |   | WHERE s.spelersnr=aantal_boetes.spelersnr<br>AND s.spelersnr=aantal_teams.spelersnr<br>ORDER BY 1,2,3,4,5;  |   |
| 2023 | Databanken:<br>Subq 2               | Geef een lijst met het spelersnummer en de naam van de spelers die in Rijswijk wonen en die in 1980 een boete gekregen hebben van 25 euro (meerdere voorwaarden dus). Gebruik hiervoor geen exists operator maar wel zijn tegenhanger die meestal bij niet-gecorreleerde subquery's wordt gebruikt. Sorteer van voor naar achter, oplopend.   | SELECT spelersnr, naam<br>FROM spelers<br>WHERE plaats = 'Rijswijk'<br>AND spelersnr in (SELECT spelersnr<br>FROM boetes<br>WHERE bedrag = 25<br>AND extract(year from datum) = 1980)<br>ORDER BY 1,2   |    |
| 2023 | Databanken:<br>Subq 2               | Geef alle spelers voor wie meer boetes zijn betaald dan dat ze wedstrijden hebben gespeeld. Zorg dat spelers zonder wedstrijd ook getoond worden. Sorteer van voor naar achter, oplopend.   | SELECT s.naam, s.voorletters, s.geb_datum<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr, s.naam, s.voorletters, s.geb_datum<br>HAVING count(*) ><br>(<br>SELECT count(*)<br>FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr)<br>ORDER BY 1,2,3  |    |
| 2023 | Databanken:<br>Subq 2               | Geef alle spelers die geen wedstrijd voor team 1 heeft gespeeld. Sorteer op naam, daarna op nr.   | SELECT spelersnr, naam<br>FROM spelers<br>WHERE spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden<br>WHERE teamnr = 1)<br>ORDER BY 2,1;   |    |
| 2023 | Databanken:<br>Subq 2               | Geef voor elke speler die ooit een boete heeft betaald, de hoogste boete weer en hoelang het geleden is dat deze boete werd betaald. Sorteer van groot naar klein op bedrag en daarna omgekeerd op "leeftijd.." van de boete.   | SELECT sub.spelersnr, sub.bedrag, age(b.datum)<br>FROM (<br>SELECT s.spelersnr, max(b.bedrag) AS bedrag<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr) AS sub<br>INNER JOIN boetes b USING (spelersnr)<br>WHERE sub.bedrag = b.bedrag<br>ORDER BY 2 DESC, 3   |    |
| 2023 | Databanken:<br>Subq 2               | Welke spelers hebben voor alle teams gespeeld uit de teamstabel ?<br>(= voor welke speler bestaat er geen enkel team waar de betreffende speler nooit voor gespeeld heeft). Sorteer op spelers nummer. Gebruik de exists operator.  | SELECT s.spelersnr<br>FROM spelers s<br>WHERE NOT EXISTS<br>(<br>SELECT '1'<br>FROM teams t<br>WHERE NOT EXISTS<br>(<br>SELECT '1'<br>FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr<br>AND w.teamnr = t.teamnr))<br>ORDER BY 1  |   |
| 2023 | Databanken:<br>Subq 2               | Maak een lijst met de spelers (naam van de speler, voorletter en woonplaats) die ooit gespeeld hebben voor een team dat nu in de tweede divisie speelt en waarvoor geen enkele boete betaald werd voor 1 januari 1981. Sorteer van voor naar achter, oplopend. Zorg dat er geen dubbels worden getoond.   | SELECT DISTINCT s.naam, s.voorletters, s.plaats<br>FROM spelers s INNER JOIN wedstrijden w USING (spelersnr)<br>WHERE w.teamnr IN<br>(<br>SELECT teamnr<br>FROM teams<br>WHERE divisie = 'tweede')<br>AND s.spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM boetes<br>WHERE datum < '1-1-1981')<br>ORDER BY 1,2,3;                            |  |
| 2023 | Databanken:<br>Subq 2               | Geef de twee laagste bondnrs terug. (tip: dwz er zijn dus minder dan 2 bondsnr die kleiner zijn). Sorteer op bondnr. Zonder het gebruik van LIMIT.  | SELECT s.bondsnr<br>FROM spelers s<br>WHERE 2 ><br>(<br>SELECT count(*)<br>FROM spelers sub<br>WHERE sub.bondsnr < s.bondsnr)<br>AND s.bondsnr IS NOT NULL<br>ORDER BY 1;   |  |
| 2023 | Databanken:<br>Subq 2<br>verbreding | Geef de diameter van het grootste hemellichaam dat bezocht is op de vroegste reis waar klantnr 126 niet op meegegaan is.  | SELECT MAX(diameter) AS grootste<br>FROM hemelobjecten<br>INNER JOIN bezoeken USING (objectnaam)<br>INNER JOIN reizen USING (reisnr)<br>WHERE vertrekdatum = (<br>SELECT MIN(vertrekdatum)<br>FROM reizen<br>INNER JOIN bezoeken USING (reisnr)<br>WHERE<br>reisnr NOT IN (<br>SELECT reisnr<br>FROM deelnames<br>WHERE klantnr = 126<br>)<br>) |  |
| 2023 | Databanken:<br>Subq 2<br>verbreding | Geef de deelnemers waarbij hun aantal reizen die ze ondernemen groter is dan alle hemelobjecten (die niet beginnen met de letter 'M') hun aantal keren dat ze bezocht zijn. Of anders geformuleerd: Geef de deelnemers met meer deelnames dan het grootste aantal bezoeken aan een hemelobject dat niet met de letter 'M' begint (:deze deelnemer meer deelnames heeft dan de "grootste" .. = deze deelnemer heeft meer deelnames dan "alle" ..)<br>Sorteer op klantnr. | SELECT klantnr, vnaam, naam, COUNT(*) AS<br>aantaldeelnames<br>FROM klanten<br>INNER JOIN deelnames using (klantnr)<br>GROUP BY klantnr, vnaam, naam<br>HAVING COUNT(*) > ALL (<br>SELECT COUNT(*)<br>FROM bezoeken<br>WHERE objectnaam NOT LIKE 'M%'<br>GROUP BY objectnaam<br>)<br>ORDER BY klantnr,2,3,4;                                    |  |





|      |                                     |  |   |   |
|------|-------------------------------------|--|---|---|
| 2023 | Databanken:<br>Subq 2<br>verbreding | Geef alle niet-bezochte hemelobjecten, buiten het grootste hemellichaam.<br>Sorteer op diameter en objectnaam.   | SELECT objectnaam, afstand, diameter<br>FROM hemelobjecten<br><br>WHERE diameter < ANY (<br>SELECT diameter<br>FROM hemelobjecten rest<br>WHERE rest.objectnaam <> hemelobjecten.objectnaam<br>)<br><br>AND NOT EXISTS (<br>SELECT *<br>FROM bezoeken<br>WHERE bezoeken.objectnaam =<br>hemelobjecten.objectnaam<br>)<br>ORDER BY diameter, objectnaam;   |    |
| 2023 | Databanken:<br>Subq 2<br>verbreding | Maak een lijst van klanten die meer dan 2 keer een reis gemaakt hebben waarbij er geen bezoek was aan Jupiter.   | SELECT k.klantnr, (k.naam    ' '    k.vnaam) AS klantnaam,<br>COUNT(d.reisnr) AS aantalreizen<br>FROM klanten k<br>INNER JOIN deelnames d USING (k.klantnr)<br>WHERE d.reisnr NOT IN<br>(SELECT b.reisnr<br>FROM bezoeken b INNER JOIN hemelobjecten o ON<br>b.objectnaam = o.objectnaam<br>WHERE o.objectnaam = 'Jupiter')<br>GROUP BY k.klantnr, k.naam, k.vnaam<br>HAVING count(d.reisnr) > 2;   |    |
| 2023 | Databanken:<br>Subq 2<br>verbreding | Geef de klantnr voor de klant met het meeste bezoeken aan de maan. Geef ook het aantal bezoeken.<br>Gebruik geen limit of top.   | SELECT d.klantnr, count(b.objectnaam)<br>FROM deelnames d<br>INNER JOIN reizen r ON d.reisnr = r.reisnr<br>INNER JOIN bezoeken b ON r.reisnr = b.reisnr<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr<br>HAVING COUNT(b.objectnaam) = (<br>SELECT MAX(c)<br>FROM (<br>SELECT d.klantnr, COUNT(b.objectnaam) AS c<br>FROM deelnames d<br>INNER JOIN bezoeken b ON d.reisnr = b.reisnr<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr<br>) AS temp<br>); |    |
| 2023 | Databanken:<br>Joins 1              | Maak een lijst met alle spelers die ooit een boete gekregen hebben die hoger is dan 50 euro. Geen dubbels. Sorteer van voor naar achter.   | SELECT DISTINCT naam, voorletters, plaats<br>FROM spelers s, boetes b<br>WHERE s.spelersnr = b.spelersnr<br>AND b.bedrag > 50<br>ORDER BY 1,2,3;  |   |
| 2023 | Databanken:<br>Joins 1              | Geef van elke boete het betalingsnr, het boetebedrag en het percentage dat het bedrag uitmaakt van de som van alle bedragen. Sorteer deze data op het betalingsnr. Zorg dat er maar twee getallen na de komma getoond worden (rond af). Sorteer van voor naar achter.  | SELECT b1.betalingsnr, b1.bedrag,<br>round(100*b1.bedrag/sum(b2.bedrag),2)<br>FROM boetes b1, boetes b2 GROUP BY b1.betalingsnr,<br>b1.bedrag ORDER BY 1,2,3;   |  |
| 2023 | Databanken:<br>Joins 1              | Geef chronologisch de bestuursleden die voorzitter zijn of geweest zijn (chronologisch op begindatum van het voorzitterschap) met vermelding van deze begindatum, alsook hun naam en huidige adres.<br>Als het vollegie adres niet gekend is dan moet "adres ongekend" weergegeven worden. Sorteer van voor naar achter.   | SELECT b.begin_datum, s.naam, coalesce(s.straat  <br>'[s.huisnr]  '[s.plaats]  '[s.postcode,'adres niet gekend')<br>AS adres<br>FROM spelers s INNER JOIN bestuursleden b<br>USING(spelersnr)<br>WHERE b.functie = 'Voorzitter'<br>ORDER BY 1,2,3;  |  |
| 2023 | Databanken:<br>Joins 1              | Geef alle wedstrijden van het team waarvan speler 6 aanvoerder is. Sorteer   | SELECT wedstrijdnr<br>FROM wedstrijden<br>WHERE TEAMNR IN<br>(SELECT teamnr<br>FROM TEAMS<br>WHERE spelersnr=6)<br>ORDER BY 1;  |  |
| 2023 | Databanken:<br>Joins 1              | Geef alle spelers die geen enkele wedstrijd voor team 1 hebben gespeeld. Sorteer op naam, daarna op spelersnr.   | SELECT spelersnr, naam<br>FROM spelers<br>WHERE spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden<br>WHERE teamnr = 1)<br>ORDER BY 2,1;   |  |
| 2023 | Databanken:<br>Joins 1              | Maak een lijst met de spelers (naam van de speler, voorletter en woonplaats) die ooit gespeeld hebben voor een team dat nu in de tweede divisie speelt en waarvoor geen enkele boete betaald werd voor 1 januari 1981. Geen dubbels, sorteer van voor naar achter.   | SELECT DISTINCT s.naam, s.voorletters, s.plaats<br>FROM spelers s NATURAL INNER JOIN wedstrijden w<br>WHERE w.teamnr IN<br>(SELECT t.teamnr<br>FROM teams t<br>WHERE divisie = 'tweede')<br>AND NOT EXISTS<br>(SELECT 2<br>FROM boetes b<br>WHERE b.datum < '1-1-1981'<br>AND b.spelersnr = s.spelersnr)<br>ORDER BY 1,2,3;   |  |
| 2023 | Databanken:<br>Joins extra          | Hoeveel kilometers heeft iedereen in totaal gevlogen tot nu toe en hoeveel hebben ze hier in totaal voor betaald. Vermits we de posities van de planeten niet kennen, mag je de afstanden van de hemelobjecten direct gebruiken. Geef het totaal gependeerde bedrag, de afgelegde kilometers, de prijs per kilometer en datum van hun laatste vlucht van al hun persoonlijke reizen. In het geval dat iemand niet op reis is geweest of geen kilometers gedaan heeft, toon je de boodschap 'veel geld voor niks of niet op reis geweest' in de kolom prijs_per_kilometer.<br>Sorteer van voor naar achter. | SELECT k.klantnr, k.naam    ' '    k.vnaam AS naam,<br>tot_bedrag, tot_afstand,<br>CASE<br>WHEN tot_afstand <> 0 THEN<br>CAST(tot_bedrag/tot_afstand AS varchar)<br>ELSE 'veel geld voor niks of niet op reis geweest'<br>END AS prijs_per_kilometer,<br>laatste_reis_datum<br>FROM klanten k<br>NATURAL LEFT OUTER JOIN<br>((SELECT klantnr, sum(prijs) AS tot_bedrag,<br>max(vertrekdatum) AS laatste_reis_datum  |  |











|      |                         |  |   |   |
|------|-------------------------|--|---|---|
|      |                         | <p>(tip frm in deze oefening: effect analoog aan cartesisch product bij het het joinen van teveel tabellen, in een andere context, nl deze van tennis: je verbindt spelers, teams en boetes, er is 1 kapitein met 2 boetes; als je hier geen aandacht voor hebt, van hoeveel teams is deze dan schijnbaar kapitein als je niet oplet..)</p>  | FROM deelnames NATURAL LEFT OUTER JOIN reizen GROUP BY klantnr) AS dr NATURAL LEFT OUTER JOIN (SELECT klantnr, sum(afstand) AS tot_afstand FROM (deelnames NATURAL INNER JOIN bezoeken) NATURAL INNER JOIN hemelobjecten GROUP BY klantnr) AS bh) ORDER BY 1,2,3,4,5,6;   |   |
| 2023 | Databanken: Joins extra | <p>Geef de volledige frequentietabel voor de diameters van de hemelobjecten (frequentie: hoeveel ojecten zijn er met de gegeven diameter, cumulatieve Frequentie, relatieve frequentie, Relatieve cumulatieve frequentie). Let op de datatypes en de precisie, gebruik CAST, rond niet af. (NUMERIC(5,2)) Sorteer op diameter en verder op de volgende kolommen.</p>   | SELECT b.diameter AS diameter, f, count(*) as "cf", CAST(f*100.00/tot AS NUMERIC(5,2))AS rf, CAST(100.00*count(*)/tot AS NUMERIC(5,2)) as "cr" FROM (SELECT diameter, count(*) AS f FROM hemelobjecten GROUP BY diameter) AS b INNER JOIN hemelobjecten bo ON (b.diameter >= bo.diameter), (SELECT count(*) as tot FROM hemelobjecten) AS boe GROUP BY b.diameter, f, tot ORDER BY 1,2,3,4,5;   |    |
| 2023 | Databanken: Joins extra | <p>Geef voor elke reis het aantal klanten waarvan de naam niet met een 'G' begint en waarvan de periode van de geboortedatum van de klant tot de vertrekdatum van de reis overlapt met de huidige datum en 50 jaar verder (gebruik hiervoor de gepaste operator: OVERLAPS). Indien er op de reis hemelobjecten worden bezocht waarvan de tweede letter van het hemelobject voorkomt in de naam van het hemelobject waarvan dit bezocht hemelobject een satelliet is, dan wordt deze reis genegeerd. Sorteer op reisnr en verder.</p> <p>(tip: conditie in outer join is niet hetzelfde als conditie in de where; + interval)</p> | SELECT r.reisnr, count(k.klantnr) FROM (REIZEN r NATURAL LEFT OUTER JOIN deelnames) LEFT OUTER JOIN klanten k ON (k.klantnr = deelnames.klantnr AND upper(naam) NOT LIKE 'G%' AND (geboortedatum,vertrekdatum) OVERLAPS (CURRENT_DATE,CURRENT_DATE+interval '50 year')) WHERE r.reisnr NOT IN (SELECT reisnr FROM bezoeken NATURAL INNER JOIN hemelobjecten WHERE lower(satellietvan) LIKE '%'    lower(substring(objectnaam FROM 2 FOR 1))    '%') GROUP BY r.reisnr ORDER BY 1,2;   |    |
| 2023 | Databanken: Set oper    | <p>Geef een overzicht van alle spelers, gevolgd door alle bestuursleden, gesorteerd op jaar van toetreding of beginjaar van hun functie en vervolgens op spelersnr. Geen dubbels tonen.</p>  | SELECT spelersnr AS veld1, naam AS veld2, jaartoe AS veld3 FROM spelers UNION SELECT spelersnr AS veld1, functie AS veld2, EXTRACT (YEAR FROM begin_datum) AS veld3 FROM bestuursleden ORDER BY veld3, veld1  |    |
| 2023 | Databanken: Set oper    | <p>Geef een lijst met alle spelersnrs, naam en het aantalwedstrijden ze gespeeld hebben en op een nieuwe lijn het aantal bestuursfuncties die ze hebben/hadden. Spelers die zowel wedstrijden gespeeld hebben als bestuurslid zijn, komen dus twee keer voor in het resultaat. Sorteer op spelersnr en aantal. Geen dubbels tonen.</p>   | SELECT spelers.spelersnr AS nr, spelers.naam, COUNT(*) AS aantal FROM spelers INNER JOIN wedstrijden USING(spelersnr) GROUP BY spelers.spelersnr, spelers.naam UNION SELECT bestuursleden.spelersnr AS nr, spelers.naam, COUNT(*) AS aantal FROM bestuursleden INNER JOIN spelers USING(spelersnr) GROUP BY bestuursleden.spelersnr, spelers.naam ORDER BY nr, aantal;  |  |
| 2023 | Databanken: Set oper    | <p>Geef een overzicht van het aantal records in de vijf tabellen uit de database tennis. Je mag hiervoor elke tabel manueel tellen. Sorteer op label.</p>  | SELECT 'boetes' AS label, COUNT(*) AS aantal FROM boetes UNION SELECT 'bestuursleden' AS label, COUNT(*) AS aantal FROM bestuursleden UNION SELECT 'spelers' AS label, COUNT(*) AS aantal FROM spelers UNION SELECT 'teams' AS label, COUNT(*) AS aantal FROM teams UNION SELECT 'wedstrijden' AS label, COUNT(*) AS aantal FROM wedstrijden ORDER BY label;  |  |
| 2023 | Databanken: Set oper    | <p>Geef een overzicht van de boetebedragen, aantal gewonnen en verloren sets en aantal verschillende functies. Bekijk de output voor de manier hoe het getoond moet worden. Sorteer van links naar rechts. Tip: Het lijkt onlogisch, maar zelfs NULL krijgt een datatype en kan niet impliciet wijzigen van datatype.</p>  | SELECT SUM(bedrag) AS boetebedrag, CAST(null AS int) AS aantalgewonnen, CAST(null AS int) AS aantalverloren, CAST(null AS int) AS aantalfuncties FROM boetes UNION SELECT null AS boetebedrag, SUM(gewonnen) AS aantalgewonnen, null AS aantalverloren, NULL AS aantalfuncties FROM wedstrijden UNION SELECT null AS boetebedrag, null AS aantalgewonnen, SUM(verloren) AS aantalverloren, NULL AS aantalfuncties FROM wedstrijden UNION SELECT null AS boetebedrag, null AS aantalgewonnen, null AS aantalverloren, COUNT(DISTINCT functie) AS aantalfuncties FROM bestuursleden ORDER BY 1,2,3,4; |  |
| 2023 | Databanken: Set oper    | <p>Geef de spelersnummers die minstens één keer bestuurslid zijn geweest. Gebruik hiervoor geen JOIN, DISTINCT, GROUP BY, IN, ANY, ALL of EXISTS. Sorteer op spelersnr</p>   | SELECT spelersnr FROM spelers INTERSECT SELECT spelersnr FROM bestuursleden ORDER BY spelersnr;   |  |
| 2023 | Databanken: Set oper    | <p>Geef de spelersnummers die geen wedstrijd gespeeld hebben. Gebruik hiervoor geen JOIN, DISTINCT, GROUP BY, IN, ANY, ALL of EXISTS. Sorteer op spelersnr</p>   | SELECT spelersnr FROM spelers EXCEPT SELECT spelersnr FROM wedstrijden ORDER BY spelersnr;  |  |



|   |                         |  |  |  |
|---|-------------------------|--|--|--|
| 2023  | Databanken:<br>Set oper | <p>Geef voor de spelers die bestuurslid en/of teamkapitein zijn hun naam en een oplingsting van hun functienamen (huidig of verleden) en hun divisies waarvoor ze kapitein zijn.<br/>Sorteer op spelersnaam en naam.<br/>Gebruik geen OUTER JOIN of WHERE.</p>   | <pre>SELECT distinct spelers.naam as spelersnaam, rommel.naam FROM spelers INNER JOIN ( SELECT functie AS naam, spelerssnr FROM bestuursleden UNION SELECT divisie AS naam, spelersnr FROM teams ) AS rommel USING (spelersnr) ORDER BY 1,2;</pre>   |  |
| 2023  | Databanken:<br>Set oper | <p>Geef een lijst van alle spelers die bestuurslid geweest zijn (of nu nog zijn) en/of een boete hebben gehad hun aantal boetes en hun laatst begonnen bestuursfunctie. Zorg dat spelers die boetes gehad hebben én bestuurder zijn (geweest) twee keer voorkomen in de kolom 'data' twee verschillende waardes.<br/>Sorteer op naam, voorletters en data.<br/>Gebruik de to_char functie voor het formaat van de geboortedatum (bv 12/12/1900).</p> | <pre>SELECT naam, voorletters, to_char(geb_datum, 'DD/MM/YYYY') AS geboortedatum, data.data FROM spelers INNER JOIN ( SELECT spelersnr, 'Boetes: '    CAST(COUNT(*) AS CHARACTER(20)) AS data FROM boetes GROUP BY spelersnr UNION SELECT spelersnr, functie AS data FROM bestuursleden WHERE begin_datum = ( SELECT MAX(begin_datum) FROM bestuursleden alle WHERE alle.spelersnr = bestuursleden.spelersnr ) ) AS data ON spelers.spelersnr = data.spelersnr ORDER BY naam, voorletters, data;</pre> |  |
| <p>Korte opgave:</p> <p>Geef alle spelers.</p> <p>Toon het aantal boetes als deze speler voor 1963 geboren is (Boetes: x)<br/>Toon de tekst 'Geen boetes' als deze speler geen boetes heeft.</p> <p>Voor bestuursleden wordt ook hun laatst gestarte functie getoond.</p> <p>(Het kan zijn dat spelers 2 keer getoond worden, namelijk oude bestuursleden)</p> <p>Voor spelers waarvoor geen infodata is, wordt de tekst 'Gewone speler' getoond ipv null.</p> <p>Gebruik de to_char functie met 'DD/MM/YYYY' voor het tonen van de geboortedatum</p> <p>Sorteer op naam, voorletters, data.</p> <p>Lange opgave:</p> <p>(Een variant op een andere opgave 176)<br/>We zijn wat betreft boetes alleen geïnteresseerd als de speler geboren is voor 1963.</p> <p>Verdere uitleg:<br/>Geef een lijst van ALLE spelers die bestuurslid geweest zijn en/of een boete hebben gehad, (wat betreft boete: alleen als ze geboren zijn voor 1963) en hun laatst begonnen bestuursfunctie. Zorg ervoor dat spelers die een bestuursfunctie hebben/hadden, geboren zijn voor 1963, maar geen boete hebben gekregen, toch in het resultaat voorkomen met een extra lijn 'Geen boetes'.<br/>Zorg ervoor dat bestuursleden, onafhankelijk van het geboortjaar, met hun laatst gestarte functie getoond worden<br/>Spelers die geen bestuurslid zijn geweest, maar een boete hebben gehad, komen één keer voor in het resultaat met hun aantal boetes.<br/>Spelers die geen bestuurslid zijn geweest en geboren zijn na 1962, moeten ook één keer in het resultaat voorkomen met als data 'Gewone speler' (het kan zelfs zijn dat geen boete gekregen hebben, alle spelers moeten sowieso getoond worden).<br/>Sorteer op naam, voorletters en data (let op de hoofdletters in het veld data).</p> <p>(ps: in deze oefening zit bijna alles wat je de voorbije twee jaar gezien hebt van SQL)</p> <p>(de vorige opgave 176 was:<br/>Geef een lijst van alle spelers die bestuurslid geweest zijn (of nu nog zijn) en/of een boete hebben gehad hun aantal boetes en hun laatst begonnen bestuursfunctie. Zorg dat spelers die boetes gehad hebben én bestuurder zijn (geweest) twee keer voorkomen in de kolom 'data' twee verschillende waardes.<br/>Sorteer op naam, voorletters en data.<br/>Gebruik de to_char functie voor het formaat van de geboortedatum (bv 12/12/1900).)</p> |                         |  |  |  |
| 2023  | Databanken:<br>Extra 1  | <p>Hoeveel sets zijn er in totaal gewonnen, hoeveel sets werden in totaal verloren en welk is het uiteindelijke saldo?</p>   | <pre>SELECT sum(gewonnen) AS totaal_gewonnen, sum(verloren) AS totaal_verloren, sum(gewonnen- verloren) AS SALDO FROM wedstrijden;</pre>   |  |
| 2023  | Databanken:<br>Extra 1  | <p>Geef een totaal overzicht van alle spelers, hun boetes en de functies die ooit vervuld hebben. Elke speler moet getoond worden, als ie een eventuele boete heeft gekregen en/of een eventuele functie als bestuurslid heeft gehad dan moet dit ook getoond worden. Toon de volledige naam, het bedrag en datum van de eventuele boete en de eventuele bestuursfuncties. Sorteer van voor naar achter.</p>   | <pre>SELECT s.naam, s.voorletters, be.functie, bo.bedrag, bo.datum FROM (spelers s LEFT OUTER JOIN bestuursleden be USING (spelersnr)) LEFT OUTER JOIN boetes bo USING (spelersnr) ORDER BY 1,2,3,4,5;</pre>   |  |
| 2023  | Databanken:<br>Extra 1  | <p>Geef een overzicht van de spelers die een boete hebben gekregen, indien deze boete meer van 90 euro is toon je informatie "pijnlijk" anders "te doen". Toon de volledige naam en de categorie van de bijhorende boete. Sorteer van voor naar achter.</p>  | <pre>SELECT s.naam, s.voorletters, CASE WHEN b.bedrag &lt;= 90 THEN 'te doen' ELSE 'pijnlijk' END AS comment FROM spelers s INNER JOIN boetes b USING (spelersnr) ORDER BY 1,2,3;</pre>  |  |
| <pre>SELECT s.spelersnr, s.naam</pre>   |                         |  |  |  |










|      |                     |   |   |   |
|------|---------------------|---|---|---|
| 2023 | Databanken: Optim   | Geef een lijst met het spelersnummer en de naam van de speler die in Rijswijk wonen en die in 1980 een boete gekregen hebben van 25 euro. Sorteer van voor naar achter. Probeer gelijk of beter te doen dan "(cost=2.37..2.37 rows=1 width=68)".  | FROM spelers s INNER JOIN boetes b USING (spelersnr) WHERE extract(year from b.datum) = 1980 AND plaats = 'Rijswijk' AND bedrag=25 ORDER BY 1,2   |    |
| 2023 | Databanken: Optim   | Geef de naam en het spelersnummer van de spelers die ooit penningmeester geweest zijn van de club, die bovendien ooit een boete betaald hebben van meer dan 75 euro, en die ooit een wedstrijd gewonnen hebben met meer dan 2 sets verschil. Sorteer van voor naar achter. Probeer gelijk of beter te doen dan "Unique (cost=100.38..100.54 rows=21 width=68)". | SELECT DISTINCT s.naam, s.spelersnr FROM ((spelers s INNER JOIN bestuursleden be USING (spelersnr)) INNER JOIN boetes bo USING (spelersnr)) INNER JOIN wedstrijden w USING (spelersnr) WHERE bo.bedrag >= 75 AND be.functie = 'Penningmeester' AND be.eind_datum IS NOT NULL AND (w.gewonnen - w.verloren) > 1 ORDER BY 1,2 |    |
| 2023 | Databanken: Optim   | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding. Sorteer van voor naar achter. Probeer gelijk of beter te doen dan "Sort (cost=33.16..33.66 rows=200 width=86)"  | SELECT s.spelersnr,s.naam,s.voorletters,s.jaartoe- (SELECT avg(jaartoe) FROM spelers) AS Verschil FROM spelers s ORDER BY 1,2,3,4   |    |
| 2023 | Databanken: Optim   | Je kan per speler berekenen hoeveel boetes die speler heeft gehad en wat het totaalbedrag per speler is. Pas nu deze query aan zodat per verschillend aantal boetes wordt getoond hoe vaak dit aantal boetes voorkwam. Sorteer van voor naar achter. Probeer gelijk of beter te doen dan "Sort (cost=46.39..46.89 rows=200 width=8)".                           | SELECT a, count(a) FROM (SELECT count(bedrag) AS a FROM boetes GROUP BY spelersnr) b GROUP BY a ORDER BY 1,2  |    |
| 2023 | Databanken: Optim   | Geef van alle spelers het verschil tussen het jaar van toetreding en het geboortjaar, maar geef alleen die spelers waarvan dat verschil groter is dan 20. Sorteer van voor naar achter. Probeer zo goed of beter te doen dan "Sort (cost=17.20..17.37 rows=67 width=90)"  | SELECT s.spelersnr, s.naam, s.voorletters, (s.jaartoe - extract(year from s.geb_datum)) AS Toetredingsleeftijd FROM spelers s WHERE (s.jaartoe - extract(year from s.geb_datum)) > 20 ORDER BY 1,2,3,4  |    |
| 2023 | Databanken: Optim   | Geef alle spelers die alfabetisch (dus naam en voorletters, in deze volgorde) voor speler 8 staan. Sorteer van voor naar achter. Probeer zo goed of beter te doen dan "Sort (cost=24.31..24.47 rows=67 width=88)"   | SELECT spelersnr, naam, voorletters, geb_datum FROM spelers WHERE naam  voorletters < (SELECT naam  voorletters FROM spelers WHERE spelersnr=8) ORDER BY 1,2,3;   |    |
| 2023 | Databanken: Extra 2 | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer van voor naar achter. Toon 3 getallen na de komma, maximaal 2 voor de komma; gebruik een cast functie.  | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe- avg as numeric(5,3)) FROM spelers s, (SELECT plaats, avg(jaartoe) AS avg FROM spelers GROUP BY plaats) a WHERE a.plaats=s.plaats ORDER BY 1, 2, 3, 4   |    |
| 2023 | Databanken: Extra 2 | Toon alle mogelijke combinaties van de letters 'x' en 'y'. Tip zie handboek, voorbeeld met cijfers. (Wat is een cartesisch product?). Sorteer   | SELECT A.letter    B.letter FROM (SELECT 'x' AS letter UNION SELECT 'y') as A, (SELECT 'x' AS letter UNION SELECT 'y') as B ORDER BY 1  |  |
| 2023 | Databanken: Extra 2 | Geef de wedstrijden die door spelers zijn gespeeld die in Leuven, Rotterdam of Leiden wonen. Sorteer van voor naar achter. Geef alle (*) informatie van wedstrijden.  | SELECT w.* FROM wedstrijden w INNER JOIN spelers s USING (spelersnr) WHERE s.plaats in ('Leuven','Leiden','Rotterdam') ORDER BY 1,2,3,4,5   |  |
| 2023 | Databanken: Extra 3 | Maak een lijst met de reizen, gevolgd door de hemelobjecten wiens objectnaam met een A begint. Sorteer van voor naar achter. TIP: Gebruik CAST AS CHAR(10) voor conversies het reisnr en vertrekdatum.  | SELECT cast(r.reisnr as char(10)) AS rnr_en_naam, cast(r.vertrekdatum As char(10)) as dtm_en_naam,reisduur, prijs FROM reizen r UNION SELECT * FROM hemelobjecten WHERE objectnaam like 'A%' ORDER BY 1,2,3,4;  |  |
| 2023 | Databanken: Extra 3 | Welke reizen hebben vijf hemelobjecten als reisdoel? Hou er rekening mee dat een reis op verschillende keren hetzelfde hemelobject kan bezoeken, deze hoeven dus niet noodzakelijk verschillend te zijn. Verduidelijking: hoe vaak is de reis uit de uitvoer bezocht? Sorteer.  | SELECT r.reisnr FROM reizen r INNER JOIN bezoeken b using (reisnr) GROUP BY r.reisnr HAVING count(b.objectnaam)=5 ORDER BY 1;   |  |
| 2023 | Databanken: Extra 3 | Maak een lijst met klantgegevens van de personen die nog nooit op Phobos op bezoek geweest zijn. Maak expliciet gebruik van de 'uitgezonderd' set operator. (Is dit de meest efficiënte oplossing?) Sorteer van voor naar achter.   | SELECT k.klantnr, k.naam, k.vnaam FROM klanten k EXCEPT SELECT k.klantnr, k.naam, k.vnaam FROM ((klanten k INNER JOIN deelnames d using (k.klantnr)) INNER JOIN reizen r using(reisnr)) INNER JOIN bezoeken b using(reisnr) WHERE b.objectnaam = 'Phobos' ORDER BY 1,2,3;   |  |
| 2023 | Databanken: Extra 4 | Maak een overzicht waarbij je voor de Maan en voor Mars aangeeft hoeveel ruimtereizen één of meer keer de betreffende bestemming bezocht hebben (d.w.z. erop geland zijn). Sorteer van voor naar achter.  | SELECT b.objectnaam, count(*) FROM bezoeken b WHERE b.objectnaam in ('Maan','Mars') AND b.verblijfsduur <> 0 GROUP BY b.objectnaam order by 1,2   |  |
| 2023 | Databanken: Extra 4 | Maak een lijst van de mensen die Mars wel bezocht hebben maar lo nog niet. Sorteer van voor naar achter.  | SELECT k.klantnr, k.naam FROM ((klanten k INNER JOIN deelnames d using (k.klantnr)) INNER JOIN reizen r using(reisnr)) INNER JOIN bezoeken b using(reisnr) WHERE b.objectnaam = 'Mars' EXCEPT SELECT k.klantnr, k.naam FROM ((klanten k INNER JOIN deelnames d using (k.klantnr)) INNER JOIN reizen r using(reisnr))        |  |

|      |                            |   |  |   |
|------|----------------------------|---|--|---|
|      |                            |   | INNER JOIN bezoeken b using(reisnr)<br>WHERE b.objectnaam = 'lo'<br>order by 1,2   |   |
| 2023 | Databanken:<br>Extra 4     | Maak een overzicht waarbij je voor de Maan en voor Mars aangeeft hoeveel verschillende ruimtereizen er geweest zijn(d.w.z. erop geland zijn is voldoende). En enkel indien er meer dan 1 reis geweest is. Sorteer van voor naar achter  | SELECT b.objectnaam, count(DISTINCT b.reisnr)<br>FROM bezoeken b<br>WHERE b.objectnaam IN ('Maan','Mars')<br>GROUP BY b.objectnaam<br>HAVING count(DISTINCT b.reisnr)>1<br>order by 1,2  |    |
| 2023 | Databanken:<br>Extra 4     | Geef de klant die het meest op de maan is geweest (+het aantal). Sorteer van voor naar achter.  | SELECT d.klantnr, count(b.objectnaam)<br>FROM (deelnames d INNER JOIN reizen r USING (reisnr))<br>INNER JOIN bezoeken b USING (reisnr)<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr<br>HAVING count(b.objectnaam) =<br>(SELECT max(c)<br>FROM (SELECT d.klantnr, count(b.objectnaam) AS c<br>FROM (deelnames d INNER JOIN reizen r USING (reisnr))<br>INNER JOIN bezoeken b<br>USING (reisnr)<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr) AS temp)<br>order by 1,2 |    |
| 2023 | Databanken:<br>Extra 4     | Maak een lijst met die mensen die meer dan 2 maal een reis ondernomen hebben waarin men geen enkele satelliet van Jupiter bezoekt !. Sorteer van voor naar achter.  | SELECT k.klantnr, (k. naam    k.vnaam) AS "Volledige naam", count(d.reisnr) AS "Aantal Ondernomen Reizen"<br>FROM klanten k JOIN deelnames d USING (k.klantnr)<br>WHERE d.reisnr NOT IN<br>(SELECT b.reisnr<br>FROM bezoeken b NATURAL INNER JOIN hemelobjecten o<br>WHERE o.satellietvan = 'Jupiter')<br>GROUP BY k.klantnr, k. naam    k.vnaam<br>HAVING count(d.reisnr)>2<br>order by 1,2,3   |    |
| 2023 | Databanken:<br>Extra 5     | Geef de nummers van de wedstrijden gespeeld door een speler waarvan de naam begint met een B of D. Sorteer van voor naar achter.  | SELECT wedstrijdnr<br>FROM wedstrijden<br>WHERE spelersnr in<br>(SELECT spelersnr<br>FROM spelers<br>WHERE substring(naam from 1 for 1) in ('B','D'))<br>order by 1  |    |
| 2023 | Databanken:<br>Extra 5     | Geef van elke speler die minstens 1 wedstrijd gewonnen heeft voor team nr 1 en voor wie in totaal meer dan 100 euro aan boete betaald is, het spelersnummer, zijn naam, woonplaats en het totale boetebedrag. Sorteer van voor naar achter.   | SELECT s.spelersnr, s.naam, s.plaats, sum(b.bedrag)<br>FROM spelers s INNER JOIN boetes b on<br>(b.spelersnr=s.spelersnr)<br>WHERE s.spelersnr IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden w<br>WHERE gewonnen > verloren<br>AND teamnr=1)<br>GROUP BY s.spelersnr, s.naam, s.plaats<br>HAVING sum(b.bedrag)>100<br>order by 1,2,3,4;  |  |
| 2023 | Databanken:<br>Extra 5     | Geef de spelers (woonplaats, naam, geslacht, in volgorde van hun geslacht en naam) voor wie minstens één boete betaald werd maar die geen aanvoerder zijn van een team. Sorteer van voor naar achter. Geen dubbels.   | SELECT DISTINCT s.geslacht, s.naam, s.plaats<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>WHERE s.spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM teams)<br>order by 1,2,3   |  |
| 2023 | Databanken:<br>Extra 5     | Geef alle spelers voor wie meer boetes zijn betaald dan dat ze wedstrijden hebben gespeeld. Sorteer van voor naar achter  | SELECT s.naam, s.voorletters, s.geb_datum<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr, s.naam, s.voorletters, s.geb_datum<br>HAVING count(*) ><br>(<br>SELECT count(*)<br>FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr)<br>order by 1,2,3   |  |
| 2023 | Databanken:<br>Extra 5     | Geef de twee laagste bondsnrs terug. (tip: dwz er zijn dus minder dan 2 bondsnr die kleiner zijn) Sorteer van voor naar achter.   | SELECT s.bondsnr<br>FROM spelers s<br>WHERE 2 ><br>(<br>SELECT count(*)<br>FROM spelers sub<br>WHERE sub.bondsnr < s.bondsnr)<br>AND s.bondsnr IS NOT NULL<br>order by 1   |  |
| 2023 | Databanken:<br>Extra 5     | Geef van elke speler die enkel wedstrijden gewonnen heeft voor team nr 1 en voor wie in totaal meer dan 100 euro aan boete betaald is, het spelersnummer, zijn naam, woonplaats en het totale boetebedrag. Dit resultaat moet aflopend geordend worden op het totale boetebedrag. Sorteer van voor naar achter. | SELECT s.spelersnr, s.naam, s.plaats, sum(b.bedrag)<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>WHERE s.spelersnr IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden w<br>WHERE gewonnen > verloren<br>AND teamnr=1)<br>GROUP BY s.spelersnr, s.naam, s.plaats<br>HAVING sum(b.bedrag)>100<br>order by 1,2,3,4   |  |
| 2023 | Databanken:<br>Venster XML | Geef alle klanten waarbij de voorlaatste letter van de naam 1 van de letters uit het woord 'azerty' is. Gebruik geen OR operator, maar een andere ISO sql operator voor het vergelijken van patronen, sorteer van voor naar achter.   | select *<br>from klanten<br>where naam similar to '%[azerty]_'<br>order by 1,2,3,4;  |  |
|      | Databanken:                | Geef voor elke een klant een overzicht aan uitgaven. Hoe? Geef voor elke klant een cumulatief overzicht van prijs van de reizen waar hij aan deelgenomen heeft. De volgorde, waarin er wordt cumulatief wordt opgeteld, wordt bepaald door de vertrekdatum van  | select klantnr, reisnr, sum(prijs) OVER (partition by klantnr<br>order by vertrekdatum)  |   |





|      |                            |  |  |   |
|------|----------------------------|--|--|---|
| 2023 | Venster XML                | de reis. Sorteer van voor naar achter.<br>Tip: vergelijk deze uitvoer met de uitvoer van een query die een gesorteerd overzicht geeft van de klanten en hun deelnames aan reizen.  | from reizen natural join deelnames<br>order by 1, 2, 3;  |      |
| 2023 | Databanken:<br>Venster XML | Geef enkel de 2 voorlaatste reizen terug. De positie van reizen wordt bepaalt door het reisnummer.<br>Ter vergelijking, als je de getallen 1 t/m 10 neemt, dan is dit 8 en 9. Sorteer van voor naar achter.<br>Gebruik enkel ISO sql.  | SELECT *<br>from<br>(select * from reizen order by reisnr desc offset 1 fetch first 2 rows only) as t<br>order by 1,2,3,4;   |    |
| 2023 | Databanken:<br>Venster XML | Toon in xmlformaat de objectnamen, afstand en diameter voor objecten die rond Neptunus draaien. Geef als eerste kolom de commentaar maan. Sorteer van voor naar achter.<br>Tip: XML is gebaseerd op het datatype text.<br><br>Verwacht uitvoer:<br>xmlcomment   xmlforest<br>-----+-----<br><!--maan-->   <objectnaam>Despina</objectnaam><br><afstand>52.500</afstand><diameter>180</diameter><br><!--maan-->   <objectnaam>Galathea</objectnaam><br><afstand>62.000</afstand><diameter>150</diameter><br><!--maan-->   <objectnaam>Larissa</objectnaam><br><afstand>73.600</afstand><diameter>192</diameter><br><!--maan-->   <objectnaam>Naiad</objectnaam><br><afstand>48.000</afstand><diameter>54</diameter><br><!--maan-->   <objectnaam>NereÄ de</objectnaam><br><afstand>5517.000</afstand><diameter>240</diameter><br><!--maan-->   <objectnaam>Proteus</objectnaam><br><afstand>117.600</afstand><diameter>416</diameter><br><!--maan-->   <objectnaam>Thalassa</objectnaam><br><afstand>50.000</afstand><diameter>80</diameter><br><!--maan-->   <objectnaam>Triton</objectnaam><br><afstand>354.800</afstand><diameter>2705</diameter>   | select xmlcomment('maan'),<br>xmlforest(objectnaam,afstand,diameter)<br>from hemelobjecten where satellietvan = 'Neptunus'<br>order by cast(xmlcomment('maan') as text),<br>cast(xmlforest(objectnaam,afstand,diameter) as text);  |    |
| 2023 | Databanken:<br>Venster XML | Geef alle klanten waarbij de 1 van de middenste letters van de naam uit het woord 'qwerty' komt.<br>Gebruik geen OR operator, sorteer van voor naar achter.  | select *<br>from klanten<br>where naam similar to '%_q[wer]ty_ %'<br>order by 1,2,3,4  |    |
| 2023 | Databanken:<br>vensters    | Geef per reis incrementeel de totale verblijfsduur volgens de volgorde waarin de hemelobjecten bezocht worden. Geef ook de totale verblijfsduur van alle reizen om alles in perspectief te zetten. Sorteer op reisnr, volgnr, objectnaam, verblijfsduur, de volgende kolommen.   | SELECT reisnr, volgnr, objectnaam, verblijfsduur,<br>sum(verblijfsduur)<br>OVER(partition by reisnr order by volgnr) as inc_duur,<br>sum(verblijfsduur) over () as tot_duur<br>FROM bezoeken<br>ORDER by 1,2,3,4,5,6;  |    |
| 2023 | Databanken:<br>vensters    | Hoe lang was het geleden dat er nog een reis vertrokken was? Geef daarnaast de totale reisduur per jaar incrementeel in de tijd (hier genaamd jaar_duur). (Dus elk jaar begint aan een nieuwe som)<br>Sorteer op reisnr en de andere kolommen.   | SELECT reisnr, lag(reisnr) OVER w1 as vorig_reisnr,<br>vertrekkdatum,<br>vertrekkdatum - lag(vertrekkdatum) OVER w1 as tussen_tijd,<br>reisduur,<br>extract(year from vertrekkdatum) as jaar,<br>sum(reisduur)<br>OVER (partition by extract(year from vertrekkdatum) order by vertrekkdatum)<br>as jaar_duur<br>FROM reizen<br>WINDOW w1 as (order by vertrekkdatum)<br>ORDER BY 1,2,3,4,5,6,7; |  |
| 2023 | Databanken:<br>GROUP BY    | Geef voor elke geboorteejaar van de klanten, het aantal klanten, het kleinste klantnummer en het grootste klantnummer. Geef ook het totaal aantal klanten en het kleinste en grootste klantnummer.<br>Sorteer van voor naar achter.  | SELECT extract(year from geboortedatum),<br>count(*),min(klantnr),max(klantnr)<br>FROM klanten<br>GROUP BY ROLLUP (extract(year from geboortedatum))<br>ORDER BY 1,2,3,4;  |  |
| 2023 | Databanken:<br>GROUP BY    | We willen telkens het aantal reizen en de totale prijs van de volgende situaties. Voor elke maand van vertrek, voor elk tiental van de reisduur, voor de combinatie van maand van vertrek en tiental van de reisduur en het het totale plaatje.<br>Sorteer van voor naar achter.   | SELECT extract(month from<br>vertrekkdatum),round(reisduur/10), count(*), sum(prijs)<br>FROM reizen<br>GROUP BY CUBE (extract(month from<br>vertrekkdatum),round(reisduur/10))<br>ORDER by 1,2,3,4;  |  |
| 2023 | Databanken:<br>GROUP BY    | Geef voor de hemelobjecten een diameter groter dan 1000 het aantal satellietvan en de gemiddelde diameter per groep die aan het volgende voldoet. We zien per object zien hoeveel satellieten hieraan voldoen, daarnaast in combinatie met hetzelfde object en afstand per 100tal. Alsook een algemeen overzicht. Sorteer van voor naar achter.  | SELECT objectnaam, round(afstand/100),satellietvan,<br>count(satellietvan), avg(diameter)<br>FROM hemelobjecten<br>WHERE diameter > 1000<br>GROUP BY ROLLUP (satellietvan,<br>(objectnaam,round(afstand/100)))<br>order by 1,2,3,4,5;  |  |
| 2023 | Databanken:<br>Exameninfo  | Lees eerst dit alvorens aan het examen te beginnen.<br>Het is een open boek examen via pc, u mag dus verschillende papieren en elektronische bronnen gebruiken. Belangrijk is dat dit gebruik 'niet-interactief' is. Interactief gebruik is niet toegelaten en wordt beschouwd als fraude. Een voorbeeld van interactief gebruik is aan een derde -direct of indirect- hulp te vragen.<br>De maximumduur van het examen is 2 uur 36 minuten. Het examen zelf is voorzien om opgelost te kunnen worden op 1 uur 30 minuten.<br>Enkel antwoorden die op de gevraagde manier worden ingediend tellen; er wordt geen rekening gehouden met andere vormen van indiening. Je dient dus op de juiste manier in te dienen. Bv een query vraag in sqldropbox dient via sqldropbox te worden ingediend en niet anders.<br>Dit examen is persoonlijk per student, het gedrag of de behandeling omtrent het naleven van deze regels van andere studenten -die zich mogelijk niet aan deze regels houden- geven u niet het recht de hier gegeven regels te overtreden. U werkt strikt individueel Het examenreglement is van toepassing. Overtredingen van de hier en in het OER gegeven regels worden beschouwd als fraude.<br>Bij de minste onduidelijkheid van deze regels brengt u onmiddellijk de beschikbare lector hiervan op de hoogte.<br>De algemene richtlijnen staan op:<br><a href="https://intranet.ucll.be/nl/student/studeren-aan-ucll/pba-de-toegepaste-informatica-proximus/examen-en-evaluatie/richtlijnen-">https://intranet.ucll.be/nl/student/studeren-aan-ucll/pba-de-toegepaste-informatica-proximus/examen-en-evaluatie/richtlijnen-</a> | select 'gelezen en goedgekeurd'  |  |










|      |                                  |  |   |   |
|------|----------------------------------|--|---|---|
|      |                                  | voor-studenten-bij-online-evalueren Gelieve deze op voorhand door te nemen.<br>Het is jouw taak om te zorgen dat benodigde hardware, software en internet werken.<br>De vak specifieke invulling staat op: <a href="https://projektwerk.ucll.be/projects/db2/wiki/Alternatieve_examinatie">https://projektwerk.ucll.be/projects/db2/wiki/Alternatieve_examinatie</a><br>Gelieve deze op voorhand door te nemen.<br>Hierbij stemt u in met deze regels en geeft u te kennen deze zonder misverstanden begrepen te hebben, dit doet u door eerst bij aanvang van het examen u naam, handtekening en de woorden 'gelezen en goedgekeurd' te onderschrijven alvorens verder te werken. Dit doe je door letterlijk "select 'gelezen en goedgekeurd'" in te geven in sqldropbox. |   |   |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Maak een lijst met de wedstrijden van speler nummer 6 waarbij je aangeeft of de wedstrijd gewonnen (toon: 'gewonnen') of verloren (toon: verloren) werd door de betreffende speler.  | <pre>SELECT spelerssnr, wedstrijdnr, gewonnen, verloren, CASE WHEN gewonnen &gt; verloren THEN 'gewonnen' ELSE 'verloren' END AS resultaat FROM wedstrijden WHERE spelerssnr = 6;</pre>   |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Welke spelers hebben voor alle teams gespeeld uit de teamstabel ?<br>(= voor welke speler bestaat er geen enkel team waar de betreffende speler nooit voor gespeeld heeft). Sorteer op spelers nummer. Gebruik de exists operator.   | <pre>SELECT s.spelersnr FROM spelers s WHERE NOT EXISTS ( SELECT '1' FROM teams t WHERE NOT EXISTS ( SELECT '1' FROM wedstrijden w WHERE w.spelersnr = s.spelersnr AND w.teamnr = t.teamnr)) ORDER BY 1</pre>   |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef voor alle spelers die geen penningmeester zijn of zijn geweest alle gewonnen wedstrijden, gesorteerd op wedstrijdnummer.  | <pre>SELECT spelerssnr, wedstrijdnr FROM wedstrijden WHERE spelerssnr NOT IN ( SELECT spelersnr FROM bestuursleden WHERE functie = 'Penningmeester') AND gewonnen &gt; verloren ORDER BY wedstrijdnr;</pre>   |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef de teams en het aantal verschillende spelers dat voor dit team gespeeld heeft.<br>Sorteer op divisie.   | <pre>SELECT teams.divisie, COUNT(DISTINCT wedstrijden.spelersnr) AS aantal FROM wedstrijden, teams WHERE wedstrijden.teamnr = teams.teamnr GROUP BY teams.divisie ORDER BY teams.divisie;</pre>   |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef een lijst met de spelers die ooit bestuurslid zijn geweest (of nog steeds zijn) en niet in Den Haag of Zoetermeer wonen.<br>Bijkomend mag deze speler maximaal 2 keer in het bestuur van de club gezeteld hebben.<br>De lijst moet aflopend gesorteerd worden op het aantal maal dat de betreffende speler in het bestuur zetelde. Mensen met hetzelfde aantal keren moeten oplopend gesorteerd worden op basis van hun spelersnr.  | <pre>SELECT spelers.naam, COUNT(*) AS aantal FROM spelers, bestuursleden WHERE bestuursleden.spelersnr = spelers.spelersnr AND plaats NOT IN ('Den Haag', 'Zoetermeer') GROUP BY spelers.spelersnr, spelers.naam HAVING COUNT(*) &lt;= 2 ORDER BY aantal DESC, spelers.spelersnr;</pre>   |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef voor alle bestuursleden een overzicht met hun naam, functie, de leeftijd die ze hadden op het moment van de start van hun functie en de leeftijd die ze hadden op het moment van het einde van hun functie.<br><br>Tip: gebruik de postgresqlfuncties AGE en EXTRACT om het aantal jaar te krijgen. En CAST(...) as CHAR(2).<br><br>(Dit is een moeilijke oefening)   | <pre>SELECT spelers.spelersnr, spelers.naam, bestuursleden.functie, CAST(EXTRACT(YEAR FROM AGE(begin_datum, geb_datum)) AS CHAR(2))    ' jaar' AS leeftijd_begin, CASE WHEN eind_datum IS NULL THEN 'nog bezig' ELSE CAST(EXTRACT(YEAR FROM AGE(eind_datum, geb_datum)) AS CHAR(2))    ' jaar' END AS leeftijd_einde FROM spelers, bestuursleden WHERE spelers.spelersnr = bestuursleden.spelersnr;</pre> |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Maak een lijst met de spelers (naam van de speler, voorletter en woonplaats) die ooit gespeeld hebben voor een team dat nu in de tweede divisie speelt en waarvoor geen enkele boete betaald werd voor 1 januari 1981. Sorteer van voor naar achter, oplopend. Zorg dat er geen dubbels worden getoond.  | <pre>SELECT DISTINCT s.naam, s.voorletters, s.plaats FROM spelers s INNER JOIN wedstrijden w USING (spelersnr) WHERE w.teamnr IN ( SELECT teamnr FROM teams WHERE divisie = 'tweede') AND s.spelersnr NOT IN ( SELECT spelersnr FROM boetes WHERE datum &lt; '1-1-1981') ORDER BY 1,2,3;</pre>  |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef het gemiddeld aantal gewonnen en verloren sets per geboortejaar.<br>Sorteer op geboortejaar.  | <pre>SELECT EXTRACT(YEAR FROM geb_datum) AS geboortejaar, AVG(gewonnen) AS gewonnen, AVG(verloren) AS verloren FROM wedstrijden, spelers WHERE wedstrijden.spelersnr = spelers.spelersnr GROUP BY EXTRACT(YEAR FROM geb_datum) ORDER BY geboortejaar;</pre>   |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Je kan per speler berekenen hoeveel boetes die speler heeft gehad en wat het totaalbedrag per speler is. Pas nu deze querye aan zodat per verschillend aantal boetes wordt getoond hoe vaak dit aantal boetes voorkwam.Sorteer eerst op de eerste kolom en daarna op de tweede kolom.  | <pre>SELECT a, count(a) FROM (SELECT count(bedrag) AS a FROM boetes GROUP BY spelersnr) b GROUP BY a ORDER BY 1,2</pre>   |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef de twee laagste bondnrs terug. (tip: dwz er zijn dus minder dan 2 bondsnr die kleiner zijn). Sorteer op bondnr. Zonder het gebruik van LIMIT.   | <pre>SELECT s.bondsnr FROM spelers s WHERE 2 &gt; ( SELECT count(*) FROM spelers sub</pre>  |  |



|      |                                     |   |   |   |
|------|-------------------------------------|---|---|---|
|      |                                     |   | WHERE sub.bondsnr < s.bondsnr)<br>AND s.bondsnr IS NOT NULL<br>ORDER BY 1;  |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef spelers die in het jaar dat ze lid geworden zijn van de club reeds een boete van meer dan 50 euro gekregen hebben en de som van al deze boetes groter of gelijk is aan 100 euro. Geef buiten de voorletters en de naam van de speler ook het aantal boetes die aan bovenstaande voorwaarden voldoen. Sorteer op spelersnr.                     | SELECT spelers.voorletters, spelers.naam, COUNT(*) AS aantalboetes<br>FROM spelers, boetes<br>WHERE spelers.spelersnr = boetes.spelersnr<br>AND boetes.bedrag > 50<br>AND EXTRACT(YEAR FROM boetes.datum) =<br>spelers.jaartoe<br>GROUP BY spelers.spelersnr, spelers.voorletters, spelers.naam<br>HAVING SUM (bedrag) >= 100<br>ORDER BY spelers.spelersnr;  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef een overzicht van alle divisies waar wedstrijden voor gespeeld zijn met spelers die actief bestuurslid zijn en minstens één boete hebben. Zorg dat elke divisie maar één keer voorkomt en sorteer de lijst alfabetisch.  | SELECT DISTINCT teams.divisie<br>FROM teams<br>INNER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr<br>INNER JOIN bestuursleden ON bestuursleden.spelersnr = wedstrijden.spelersnr<br>INNER JOIN boetes ON wedstrijden.spelersnr = boetes.spelersnr<br>WHERE bestuursleden.eind_datum IS NULL<br>ORDER BY teams.divisie;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef voor de actieve bestuursleden zonder boete hun laatste gespeelde wedstrijd (die met het hoogste wedstrijdnummer). Sorteer aflopend op spelersnr.   | SELECT bestuursleden.spelersnr,<br>MAX(wedstrijden.wedstrijdnr) AS laatstewedstrijd<br>FROM bestuursleden<br>INNER JOIN wedstrijden ON bestuursleden.spelersnr = wedstrijden.spelersnr AND bestuursleden.eind_datum IS NULL<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr = boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY bestuursleden.spelersnr<br>ORDER BY spelersnr DESC;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef per team de leeftijd van de aanvoerder (tip: postgresql heeft een AGE() functie) en het aantal verschillende spelers dat voor dit team gespeeld heeft. Alleen teams waarvoor wedstrijden zijn gespeeld en die een aanvoerder hebben, moeten vermeld worden. Sorteer op leeftijd en daarna op aantal verschillende spelers en daarna op teamnr. | SELECT teams.teamnr, CAST(EXTRACT(year from AGE(NOW(), spelers.geb_datum)) AS varchar(2))    ' jaar' AS leeftijd, COUNT(DISTINCT wedstrijden.spelersnr) AS aantalspelers<br>FROM teams<br>INNER JOIN spelers ON teams.spelersnr = spelers.spelersnr<br>INNER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr<br>GROUP BY teams.teamnr, spelers.geb_datum<br>ORDER BY leeftijd, aantalspelers, teamnr;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef alle spelers die meer wedstrijden gespeeld hebben dan het aantal wedstrijden dat de huidige voorzitter heeft verloren. De huidige voorzitter komt zelf niet in de lijst voor. Gebruik geen subqueries. Sorteer op spelersnr.   | SELECT spelers.spelersnr<br>FROM spelers<br>INNER JOIN wedstrijden ON spelers.spelersnr = wedstrijden.spelersnr,<br>bestuursleden<br>INNER JOIN wedstrijden voorzitterswedstrijden ON bestuursleden.spelersnr = voorzitterswedstrijden.spelersnr<br>AND voorzitterswedstrijden.gewonnen < voorzitterswedstrijden.verloren<br>WHERE bestuursleden.eind_datum IS NULL AND bestuursleden.functie = 'Voorzitter' AND bestuursleden.spelersnr <> spelers.spelersnr<br>GROUP BY spelers.spelersnr<br>HAVING COUNT(DISTINCT wedstrijden.wedstrijdnr) > COUNT(DISTINCT voorzitterswedstrijden.wedstrijdnr)<br>ORDER BY spelers.spelersnr; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef voor elke aanvoerder het spelersnr, de naam en het aantal boetes dat voor hem of haar betaald is en het aantal teams dat hij of zij aanvoert. Aanvoerders zonder boetes mogen niet getoond worden. Sorteer, beginnend bij de eerste kolom, eindigend bij de laatste kolom.   | SELECT s.spelersnr, s.naam, s.voorletters, aantalboetes, aantalteams<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes,<br>(SELECT spelersnr, count(*) AS aantalteams<br>FROM teams<br>GROUP BY spelersnr) AS aantal_teams,<br>spelers s<br>WHERE s.spelersnr=aantal_boetes.spelersnr<br>AND s.spelersnr=aantal_teams.spelersnr<br>ORDER BY 1,2,3,4,5  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef een overzicht van het aantal records in de vijf tabellen uit de database tennis. Je mag hiervoor elke tabel manueel tellen. Sorteer op label.  | SELECT 'boetes' AS label, COUNT(*) AS aantal<br>FROM boetes<br>UNION<br>SELECT 'bestuursleden' AS label, COUNT(*) AS aantal<br>FROM bestuursleden<br>UNION<br>SELECT 'spelers' AS label, COUNT(*) AS aantal<br>FROM spelers<br>UNION<br>SELECT 'teams' AS label, COUNT(*) AS aantal<br>FROM teams<br>UNION<br>SELECT 'wedstrijden' AS label, COUNT(*) AS aantal<br>FROM wedstrijden<br>ORDER BY label;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef het totaal aantal boetes, het totale boetebedrag, het minimum en het maximum boetebedrag dat door onze club betaald werd. Let er hierbij op dat er gehele getallen worden getoond (rond af indien nodig). Sorteer van voor naar achter, oplopend.  | SELECT count(*) AS Aantal_boetes, round(sum (bedrag)) AS Totaal_bedrag, round(min (bedrag)) as Minimum, round(max (bedrag)) as Maximum<br>FROM boetes<br>ORDER BY 1,2,3,4;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef de spelers (woonplaats, naam, geslacht, in volgorde van hun geslacht en naam) voor wie minstens één boete betaald werd maar die geen aanvoerder zijn van een team. Sorteer van voor naar achter. Geen dubbels.   | SELECT DISTINCT s.geslacht, s.naam, s.plaats<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>WHERE s.spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM teams)<br>)   |  |












|      |                                  |   |  |   |
|------|----------------------------------|---|--|---|
|      |                                  |   | order by 1,2,3   |   |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Je kan per speler berekenen hoeveel boetes die speler heeft gehad en wat het totaalbedrag per speler is. Pas nu deze querye aan zodat per verschillend aantal boetes wordt getoond hoe vaak dit aantal boetes voorkwam. Sorteer van voor naar achter. Probeer gelijk of beter te doen dan "Sort (cost=46.39..46.89 rows=200 width=8)".      | SELECT a, count(a)<br>FROM (SELECT count(bedrag) AS a<br>FROM boetes<br>GROUP BY spelersnr) b<br>GROUP BY a<br>ORDER BY 1,2  |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef voor elke speler die een wedstrijd heeft gespeeld het spelersnr en het totaal aantal boetes. Spelers die een wedstrijd gespeeld hebben, maar geen boetes hebben, moeten ook getoond worden. Sorteer op het aantal boetes en op spelersnr;  | SELECT DISTINCT w.spelersnr, aantalboetes<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes<br>RIGHT OUTER JOIN wedstrijden w USING (spelersnr)<br>ORDER BY aantalboetes  |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef voor elke aanvoerder het spelersnr, de naam en het aantal boetes dat voor hem of haar betaald is en het aantal teams dat hij of zij aanvoert. Toon enkel aanvoerders die boetes gekregen hebben. Sorteer van voor naar achter, oplopend.   | SELECT s.spelersnr, s.anaam, s.voorletters, aantalboetes, aantalteams<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes,<br>(SELECT spelersnr, count(*) AS aantalteams<br>FROM teams<br>GROUP BY spelersnr) AS aantal_teams,<br>spelers s<br>WHERE s.spelersnr=aantal_boetes.spelersnr<br>AND s.spelersnr=aantal_teams.spelersnr<br>ORDER BY 1,2,3,4,5; |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef voor de spelers die bestuurslid en/of teamkapitein zijn hun naam en een oplistng van hun functienamen (huidig of verleden) en hun divisies waarvoor ze kapitein zijn. Sorteer op spelersnaam en naam. Gebruik geen OUTER JOIN of WHERE.  | SELECT distinct spelers.anaam as spelersnaam,<br>rommel.anaam<br>FROM spelers<br>INNER JOIN (<br>SELECT functie AS naam, spelersnr<br>FROM bestuursleden<br>UNION<br>SELECT divisie AS naam, spelersnr<br>FROM teams<br>) AS rommel USING (spelersnr)<br>ORDER BY 1,2;   |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer van voor naar achter. Toon 3 getallen na de komma, maximaal 2 voor de komma; gebruik een cast functie.  | SELECT s.spelersnr,s.anaam,s.voorletters,cast (s.jaartoe-<br>avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1, 2, 3, 4  |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef voor elke mannelijke speler wiens naam minstens 2 keer de letter 'e' bevat zijn functie die hij op dit moment uitoefent, als die er op dit moment één heeft. Sorteer op naam en functie.   | select naam, geslacht, functie<br>from spelers left outer join bestuursleden<br>on spelers.spelersnr = bestuursleden.spelersnr<br>and bestuursleden.eind_datum is null<br>where geslacht = 'M'<br>and naam like '%e%e%'<br>ORDER BY naam, functie;   |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer op spelersnr. Zet het berekende verschil om naar het datatype numeric met precisie 5 en schaal 3.   | SELECT s.spelersnr,s.anaam,s.voorletters,cast (s.jaartoe-<br>avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1   |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef een lijst met het spelersnummer en de naam van de spelers die in Rijswijk wonen en die in 1980 een boete gekregen hebben van 25 euro (meerdere voorwaarden dus). Gebruik hiervoor geen exists operator maar wel zijn tegenhanger die meestal bij niet-gecorreleerde subquery's wordt gebruikt. Sorteer van voor naar achter, oplopend. | SELECT spelersnr, anaam<br>FROM spelers<br>WHERE plaats = 'Rijswijk'<br>AND spelersnr in (SELECT spelersnr<br>FROM boetes<br>WHERE bedrag = 25<br>AND extract(year from datum) = 1980)<br>ORDER BY 1,2   |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef van alle spelers het verschil tussen het jaar van toetreding en het geboortjaar, maar geef alleen die spelers waarvan dat verschil groter is dan 20. Sorteer van voor naar achter. Probeer zo goed of beter te doen dan "Sort (cost=17.20..17.37 rows=67 width=90)".   | SELECT s.spelersnr, s.anaam, s.voorletters, (s.jaartoe -<br>extract(year from s.geb_datum)) AS Toetredingsleeftijd<br>FROM spelers s<br>WHERE (s.jaartoe - extract(year from s.geb_datum)) > 20<br>ORDER BY 1,2,3,4  |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef alle wedstrijden van het team waarvan speler 6 aanvoerder is. Sorteer  | SELECT wedstrijdnr<br>FROM wedstrijden<br>WHERE TEAMNR IN<br>(SELECT teamnr<br>FROM teams<br>WHERE spelersnr=6)<br>ORDER BY 1;   |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Toon alle mogelijke combinaties van de letters 'x' en 'y'. Tip zie handboek, voorbeeld met cijfers. (Wat is een cartesisch product?). Sorteer   | SELECT A.letter    B.letter<br>FROM (SELECT 'x' AS letter UNION SELECT 'y') as A,<br>(SELECT 'x' AS letter UNION SELECT 'y') as B<br>ORDER BY 1  |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef alle spelers die alfabetisch (dus naam en voorletters, in deze volgorde) voor speler 8 staan. Sorteer van voor naar achter. Probeer zo goed of beter te doen dan "Sort (cost=24.31..24.47 rows=67 width=88)".  | SELECT spelersnr, anaam, voorletters, geb_datum<br>FROM spelers<br>WHERE anaam  voorletters <<br>(SELECT anaam  voorletters<br>FROM spelers<br>WHERE spelersnr=8)<br>ORDER BY 1,2,3;   |  |
| 2023 | Databanken: DB2                  | Geef alle spelers voor wie meer boetes zijn betaald dan dat ze wedstrijden hebben gespeeld. Zorg dat spelers zonder wedstrijd ook getoond worden.   | SELECT s.anaam, s.voorletters, s.geb_datum<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr, s.anaam, s.voorletters,<br>s.geb_datum<br>HAVING count(*) ><br>(   |  |










|      |                                     |  |   |   |
|------|-------------------------------------|--|---|---|
|      | Examenvraag 1                       | Sorteer van voor naar achter, oplopend.  | SELECT count(*)<br>FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr)<br>ORDER BY 1,2,3   |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef alle spelers die geen enkele wedstrijd voor team 1 hebben gespeeld. Sorteer op naam, daarna op spelersnr.   | SELECT spelersnr, naam<br>FROM spelers<br>WHERE spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden<br>WHERE teamnr = 1)<br>ORDER BY 2,1;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef de spelersgegevens van de speler(s) met het hoogste bedrag (voor één boete, niet het totaalbedrag). Als twee spelers een even hoge boete gehad hebben, moeten beide spelers getoond worden (LIMIT is dus geen optie). Sorteer alfabetisch op naam en voorletters.   | SELECT spelers.spelersnr, voorletters, naam<br>FROM spelers<br>INNER JOIN boetes using (spelersnr)<br>WHERE boetes.bedrag = (<br>SELECT MAX(bedrag)<br>FROM boetes<br>)<br>ORDER BY naam, voorletters   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef een overzicht van de boetebedragen, aantal gewonnen en verloren sets en aantal verschillende functies. Bekijk de output voor de manier hoe het getoond moet worden. Sorteer van links naar rechts. Tip: Het lijkt onlogisch, maar zelfs NULL krijgt een datatype en kan niet impliciet wijzigen van datatype.   | SELECT SUM(bedrag) AS boetebedrag, CAST(null AS int)<br>AS aantalgewonnen, CAST(null AS int) AS aantalverloren,<br>CAST(null AS int) AS aantalfuncties<br>FROM boetes<br>UNION<br>SELECT null AS boetebedrag, SUM(gewonnen) AS<br>aantalgewonnen, null AS aantalverloren, NULL AS<br>aantalfuncties<br>FROM wedstrijden<br>UNION<br>SELECT null AS boetebedrag, null AS aantalgewonnen,<br>SUM(verloren) AS aantalverloren, NULL AS aantalfuncties<br>FROM wedstrijden<br>UNION<br>SELECT null AS boetebedrag, null AS aantalgewonnen,<br>null AS aantalverloren, COUNT(DISTINCT functie) AS<br>aantalfuncties<br>FROM bestuursleden<br>ORDER BY 1,2,3,4; |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef de twee laagste bondnrs terug. (tip: dwz er zijn dus minder dan 2 bondsnr die kleiner zijn) Sorteer van voor naar achter.   | SELECT s.bondsnr<br>FROM spelers s<br>WHERE 2 ><br>(<br>SELECT count(*)<br>FROM spelers sub<br>WHERE sub.bondsnr < s.bondsnr)<br>AND s.bondsnr IS NOT NULL<br>order by 1  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef per team de verloren wedstrijden. Zorg dat teams zonder verloren wedstrijden ook in de output verschijnen. Duid per wedstrijd aan of het om een actief bestuurslid gaat. Sorteer op divisie en wedstrijdnummer.   | SELECT teams.teamnr, teams.divisie,<br>wedstrijden.wedstrijdnr, wedstrijden.spelersnr,<br>CASE<br>WHEN bestuursleden.spelersnr IS NULL THEN ''<br>ELSE 'actief'<br>END AS bestuurslid<br>FROM teams<br>LEFT OUTER JOIN wedstrijden ON teams.teamnr =<br>wedstrijden.teamnr AND wedstrijden.verloren ><br>wedstrijden.gewonnen<br>LEFT OUTER JOIN bestuursleden ON<br>wedstrijden.spelersnr = bestuursleden.spelersnr AND<br>bestuursleden.eind_datum IS NULL<br>ORDER BY divisie, wedstrijden.wedstrijdnr;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef alle spelers die geen wedstrijd voor team 1 heeft gespeeld. Sorteer op naam, daarna op nr.  | SELECT spelersnr, naam<br>FROM spelers<br>WHERE spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden<br>WHERE teamnr = 1)<br>ORDER BY 2,1;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Maak een lijst met de spelers (naam van de speler, voorletter en woonplaats) die ooit gespeeld hebben voor een team dat nu in de tweede divisie speelt en waarvoor geen enkele boete betaald werd voor 1 januari 1981. Geen dubbels, sorteer van voor naar achter.   | SELECT DISTINCT s.naam, s.voorletters, s.plaats<br>FROM spelers s NATURAL INNER JOIN wedstrijden w<br>WHERE w.teamnr IN<br>(SELECT t.teamnr<br>FROM teams t<br>WHERE divisie = 'tweede')<br>AND NOT EXISTS<br>(SELECT 2<br>FROM boetes b<br>WHERE b.datum < '1-1-1981'<br>AND b.spelersnr = s.spelersnr)<br>ORDER BY 1,2,3;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | We willen een statistiek van hoeveel wedstrijden de spelers winnen. Geef een lijst met aantal gewonnen wedstrijden en het aantal spelers dat dit aantal wedstrijden gewonnen heeft. Dus bv als er vier spelers zijn die elk drie wedstrijden hebben gewonnen, dan is de output: aantal_gewonnen: 3, aantal_spelers: 4. Dit graag voor alle aantallen gewonnen wedstrijden en alle spelers. Sorteer op aantal gewonnen wedstrijden. | SELECT aantal_gewonnen, count(spelersnr) AS<br>aantal_spelers<br>FROM (<br>SELECT spelersnr, count(*) AS aantal_gewonnen<br>FROM wedstrijden<br>WHERE gewonnen > verloren<br>GROUP BY spelersnr) AS gewonnen<br>GROUP BY aantal_gewonnen<br>ORDER BY aantal_gewonnen;   |  |
| 2023 | Databanken:<br>DB2                  | Geef alle spelers die meer wedstrijden gespeeld hebben dan het aantal wedstrijden dat de huidige voorzitter heeft verloren. De huidige voorzitter komt zelf niet in de lijst voor. Gebruik geen  | SELECT spelers.spelersnr<br>FROM spelers<br>INNER JOIN wedstrijden ON spelers.spelersnr =<br>wedstrijden.spelersnr,<br>bestuursleden<br>INNER JOIN wedstrijden voorzitterswedstrijden ON<br>bestuursleden.spelersnr = voorzitterswedstrijden.spelersnr<br>AND voorzitterswedstrijden.gewonnen <   |  |










|      |                                     |   |  |   |
|------|-------------------------------------|---|--|---|
|      | Examenvraag 1                       | subqueries.<br>Sorteer op spelersnr.  | voorzitterswedstrijden.verloren<br>WHERE bestuursleden.eind_datum IS NULL AND<br>bestuursleden.functie = 'Voorzitter' AND<br>bestuursleden.spelersnr <> spelers.spelersnr<br>GROUP BY spelers.spelersnr<br>HAVING COUNT(DISTINCT wedstrijden.wedstrijdnr) ><br>COUNT(DISTINCT voorzitterswedstrijden.wedstrijdnr)<br>ORDER BY spelers.spelersnr;   |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer op spelersnr. Toon 3 getallen na de komma, zet het verschil om naar het numeric type met precisie van 5 en een schaal van 3.  | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe-<br>avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef voor elke speler die ooit een boete heeft betaald, de hoogste boete weer en hoelang het geleden is dat deze boete werd betaald. Sorteer van groot naar klein op bedrag en daarna omgekeerd op "leeftijd.." van de boetes.  | SELECT sub.spelersnr, sub.bedrag, age(b.datum)<br>FROM (<br>SELECT s.spelersnr, max(b.bedrag) AS bedrag<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr) AS sub<br>INNER JOIN boetes b USING (spelersnr)<br>WHERE sub.bedrag = b.bedrag<br>ORDER BY 2 DESC, 3  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef per team het hoogste wedstrijdnummer van een wedstrijd, gespeeld door een bestuurslid (actief en niet meer actief) die geen boete heeft gekregen. Sorteer op teamnr.   | SELECT teams.teamnr, MAX(wedstrijden.wedstrijdnr) AS<br>laatstewedstrijd<br>FROM teams<br>INNER JOIN wedstrijden on teams.teamnr =<br>wedstrijden.teamnr<br>INNER JOIN bestuursleden ON wedstrijden.spelersnr =<br>bestuursleden.spelersnr<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr =<br>boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY teams.teamnr<br>ORDER BY teams.teamnr;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef een lijst met alle spelersnrs, naam en het aantalwedstrijden ze gespeeld hebben en op een nieuwe lijn het aantal bestuursfuncties die ze hebben/hadden. Spelers die zowel wedstrijden gespeeld hebben als bestuurslid zijn, komen dus twee keer voor in het resultaat. Sorteer op spelersnr en aantal. Geen dubbels tonen.   | SELECT spelers.spelersnr AS nr, spelers.naam,<br>COUNT(*) AS aantal<br>FROM spelers<br>INNER JOIN wedstrijden USING(spelersnr)<br>GROUP BY spelers.spelersnr, spelers.naam<br>UNION<br>SELECT bestuursleden.spelersnr AS nr, spelers.naam,<br>COUNT(*) AS aantal<br>FROM bestuursleden<br>INNER JOIN spelers USING(spelersnr)<br>GROUP BY bestuursleden.spelersnr, spelers.naam<br>ORDER BY nr, aantal;  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef alle spelers voor wie meer boetes zijn betaald dan dat ze wedstrijden hebben gespeeld. Sorteer van voor naar achter  | SELECT s.naam, s.voorletters, s.geb_datum<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr, s.naam, s.voorletters,<br>s.geb_datum<br>HAVING count(*) ><br>(<br>SELECT count(*)<br>FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr)<br>order by 1,2,3  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef een lijst van alle spelers die bestuurslid geweest zijn (of nu nog zijn) en/of een boete hebben gehad hun aantal boetes en hun laatst begonnen bestuursfunctie. Zorg dat spelers die boetes gehad hebben én bestuurder zijn (geweest) twee keer voorkomen in de kolom 'data' twee verschillende waardes. Sorteer op naam, voorletters en data. Gebruik de to_char functie voor het formaat van de geboortedatum (bv 12/12/1900). | SELECT naam, voorletters, to_char(geb_datum,<br>'DD/MM/YYYY') AS geboortedatum, data.data<br>FROM spelers<br>INNER JOIN (<br>SELECT spelersnr, 'Boetes: '    CAST(COUNT(*) AS<br>CHARACTER(20)) AS data<br>FROM boetes<br>GROUP BY spelersnr<br>UNION<br>SELECT spelersnr, functie AS data<br>FROM bestuursleden<br>WHERE begin_datum = (<br>SELECT MAX(begin_datum)<br>FROM bestuursleden alle<br>WHERE alle.spelersnr = bestuursleden.spelersnr<br>)<br>) AS data ON spelers.spelersnr = data.spelersnr<br>ORDER BY naam, voorletters, data; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef voor elke mannelijke speler wiens naam minstens 2 keer de letter 'e' bevat een lijst van de functies die hij op dit moment uitoefent. Ook mannelijke spelers zonder huidige functie moeten getoond worden. Sorteer op spelersnr.   | select naam, geslacht, functie<br>from spelers left outer join bestuursleden<br>on spelers.spelersnr = bestuursleden.spelersnr<br>and bestuursleden.eind_datum is null<br>where geslacht = 'M'<br>and naam like '%e%e%'<br>ORDER BY spelers.spelersnr;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef het spelersnummer en bondsnummer van alle spelers die jonger zijn dan de speler met bondsnummer 8467. Sorteer op spelersnr   | SELECT S.spelersnr, S.bondsnr<br>FROM spelers as S INNER JOIN spelers as P<br>ON S.geb_datum > P.geb_datum<br>WHERE P.bondsnr = '8467'<br>ORDER BY S.spelersnr;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef een lijst met spelers (naam, voorletters en geboortedatum (to_char)) die nog geen boete gekregen hebben met een bedrag hoger dan 75 euro. Sorteer op spelersnr.  | SELECT spelers.naam, spelers.voorletters,<br>TO_CHAR(spelers.geb_datum, 'DD/MM/YYYY') AS<br>geboortedatum<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr =<br>boetes.spelersnr AND boetes.bedrag > 75<br>WHERE boetes.betalingsnr IS NULL<br>ORDER BY spelers.spelersnr;   |  |
|      |                                     |   | SELECT S.spelersnr, S.naam, B.functie, B.begin_datum,<br>W.wedstrijdnr<br>FROM spelers S   |   |

|      |                                     |  |  |  |
|------|-------------------------------------|--|--|--|
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef een lijst van alle huidige bestuursleden die nog geen boete gekregen hebben, maar wel al minstens één wedstrijd verloren hebben.<br>Sorteer op spelersnr en wedstrijdnr.  | INNER JOIN bestuursleden B ON S.spelersnr = B.spelersnr AND B.eind_datum IS NULL<br>INNER JOIN wedstrijden W ON S.spelersnr = W.spelersnr AND W.gewonnen < W.verloren<br>LEFT OUTER JOIN boetes BO ON S.spelersnr = BO.spelersnr<br>WHERE BO.betalingsnr IS NULL<br>ORDER BY S.spelersnr, W.wedstrijdnr  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef het gemiddeld boetebedrag per speler, afgerond op twee cijfers na de komma. spelers zonder boete moeten bovenaan staan met 'geen boetes'<br>Sorteer daarna op spelersnaam en gemiddeld boetebedrag.<br>Om een waarde om te zetten naar een ander datatype, kan je ofwel CAST() gebruiken ofwel de TO_CHAR() met 'FM999.00' als opmaakmasker.  | SELECT spelers.naam,<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 'geen boetes'<br>ELSE CAST(ROUND(AVG(boetes.bedrag), 2) AS<br>varchar(8))<br>END<br>AS gemiddeld<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr = boetes.spelersnr<br>GROUP BY spelers.spelersnr, spelers.naam<br>ORDER BY<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 0<br>ELSE 1<br>END, spelers.naam, AVG(boetes.bedrag);  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef per team de verloren wedstrijden. Zorg dat teams zonder verloren wedstrijden ook in de output verschijnen.<br>Duid per wedstrijd aan of het om een actief bestuurslid gaat.<br>Sorteer op divisie en wedstrijdnummer.   | SELECT teams.teamnr, teams.divisie,<br>wedstrijden.wedstrijdnr, wedstrijden.spelersnr,<br>CASE<br>WHEN bestuursleden.spelersnr IS NULL THEN ''<br>ELSE 'actief'<br>END AS bestuurslid<br>FROM teams<br>LEFT OUTER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr AND wedstrijden.verloren > wedstrijden.gewonnen<br>LEFT OUTER JOIN bestuursleden ON wedstrijden.spelersnr = bestuursleden.spelersnr AND bestuursleden.eind_datum IS NULL<br>ORDER BY divisie, wedstrijden.wedstrijdnr;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef een lijst van alle spelers (spelersnr en woonplaats) die met minstens twee in dezelfde plaats wonen. Sorteer aflopend op woonplaats, daarna op spelersnr.   | SELECT spelersnr, plaats<br>FROM spelers<br>WHERE plaats IN (<br>SELECT plaats<br>FROM spelers<br>GROUP BY plaats<br>HAVING COUNT(*) >= 2<br>)<br>ORDER BY plaats DESC, spelersnr;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef de naam en het spelersnummer van de spelers die ooit penningmeester geweest zijn van de club, die bovendien ooit een boete betaald hebben van meer dan 75 euro, en die ooit een wedstrijd gewonnen hebben met meer dan 2 sets verschil. Sorteer van voor naar achter.<br>Probeer gelijk of beter te doen dan "Unique (cost=100.38..100.54 rows=21 width=68)".   | SELECT DISTINCT s.naam, s.spelersnr<br>FROM ((spelers s INNER JOIN bestuursleden be USING (spelersnr)) INNER JOIN boetes bo USING (spelersnr))<br>INNER JOIN wedstrijden w USING (spelersnr)<br>WHERE bo.bedrag >= 75<br>AND be.functie = 'Penningmeester'<br>AND be.eind_datum IS NOT NULL<br>AND (w.gewonnen - w.verloren) > 1<br>ORDER BY 1,2   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding. Sorteer van voor naar achter.<br>Probeer gelijk of beter te doen dan "Sort (cost=33.16..33.66 rows=200 width=86)".   | SELECT s.spelersnr,s.naam,s.voorletters,s.jaartoe-<br>(SELECT avg(jaartoe)<br>FROM spelers) AS Verschil<br>FROM spelers s<br>ORDER BY 1,2,3,4  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | Geef een overzicht van alle boetes, onderverdeeld in categorie:<br><br>LAAG: boetebedrag tussen 0 en 40 (kleiner dan 40)<br>MIDDELMATIG: tussen 40 en 80 (kleiner dan 80)<br>HOOG: meer of gelijk aan 80<br><br>In overzicht toon je betalingsnr, spelersnr, spelersnaam, bedrag en categorie van de boete<br><br>Sorteer op naam van de speler, en op de volgende kolommen  | select naam, spelers.spelersnr, betalingsnr, bedrag,<br>case when bedrag < 40 then 'laag'<br>when bedrag < 80 then 'middelmatic'<br>else 'hoog'<br>end as categorie<br>from spelers, boetes<br>where spelers.spelersnr = boetes.spelersnr<br>order by naam, 2, 3, 4, 5   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 1 | We zijn wat betreft boetes alleen geïnteresseerd als de speler geboren is voor 1963.<br><br>Verder uitleg:<br>Geef een lijst van ALLE spelers die bestuurslid geweest zijn en/of een boete hebben gehad, (alleen als ze geboren zijn voor 1963) en hun laatst begonnen bestuursfunctie.<br>Zorg ervoor dat spelers die een bestuursfunctie hebben/hadden, geboren zijn voor 1963, maar geen boete hebben gekregen, toch in het resultaat voorkomen met een extra lijn 'Geen boetes'.<br>Spelers die geen bestuurslid zijn geweest, maar een boete hebben gehad, komen één keer voor in het resultaat met hun aantal boetes.<br>Spelers die geen bestuurslid zijn geweest en geboren zijn na 1962, moeten ook één keer in het resultaat voorkomen met als data 'Gewone speler' (het kan zelfs zijn dat geen boete gekregen hebben, alle spelers moeten sowieso getoond worden).<br>Sorteer op naam, voorletters en data (let op de hoofdletters in het veld data).<br><br>Gebruik de to_char functie voor het formaat van de geboortedatum (bv 12/12/1900). | SELECT naam, voorletters, to_char(geb_datum,<br>'DD/MM/YYYY') AS geboortedatum,<br>CASE<br>WHEN data.data IS NULL THEN 'Gewone speler'<br>ELSE data.data<br>END AS data<br>FROM spelers<br>LEFT OUTER JOIN (<br>SELECT spelers.spelersnr,<br>CASE<br>WHEN COUNT(betalingsnr) = 0 THEN 'Geen boetes'<br>ELSE 'Boetes: '    CAST(COUNT(betalingsnr) AS<br>CHARACTER(20))<br>END AS data<br>FROM spelers<br>LEFT OUTER JOIN boetes USING (spelersnr)<br>WHERE EXTRACT(YEAR FROM geb_datum) < 1963<br>GROUP BY spelersnr<br>UNION<br>SELECT spelersnr, functie AS data<br>FROM bestuursleden<br>WHERE begin_datum = (<br>SELECT MAX(begin_datum)<br>FROM bestuursleden alle<br>WHERE alle.spelersnr = bestuursleden.spelersnr<br>)<br>) AS data ON spelers.spelersnr = data.spelersnr<br>ORDER BY naam, voorletters, data; |  |
|      |                                     |  | SELECT DISTINCT wedstrijden.spelersnr,<br>boetes.betalingsnr, TO_CHAR(boetes.datum,  |  |



|      |                                  |   |   |   |
|------|----------------------------------|---|---|---|
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef alle boetes groter dan 25 euro voor aanvoerders die meegespeeld hebben in een winnende wedstrijd met het team waarvan ze aanvoerder zijn.<br>Sorteer op datum van de boete (de tekstuele waarde (to_char), niet de echte datum).   | 'DD/MM/YYYY') AS boetedatum<br>FROM teams, wedstrijden, boetes<br>WHERE wedstrijden.spelersnr = teams.spelersnr<br>AND teams.teamnr = wedstrijden.teamnr<br>AND wedstrijden.spelersnr = boetes.spelersnr<br>AND boetes.bedrag > 25<br>AND wedstrijden.gewonnen > wedstrijden.verloren<br>ORDER BY boetedatum;   |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef van elke boete het betalingsnr, het boetebedrag en het percentage dat het bedrag uitmaakt van de som van alle bedragen.<br>Sorteer deze data op het betalingsnr. Zorg dat er maar twee getallen na de komma getoond worden (rond af). Sorteer van voor naar achter.  | SELECT b1.betalingsnr, b1.bedrag,<br>round (100*b1.bedrag/sum(b2.bedrag),2)<br>FROM boetes b1, boetes b2 GROUP BY b1.betalingsnr,<br>b1.bedrag ORDER BY 1,2,3;  |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef alle bestuursleden die geboren zijn voor 1970-08-05 en hun staat van dienst.<br>Bekijk de output om de juiste formatering te zien.<br>gebruik een combinatie van CASE, de    functie en datum opmaak.<br>Sorteer op naam en begin datum  | SELECT spelers.voorletters, spelers.naam,<br>bestuursleden.functie, 'actief: '   <br>CASE<br>WHEN bestuursleden.eind_datum IS NULL THEN 'sinds '<br>   to_char(bestuursleden.begin_datum, 'DD/MM/YYYY')<br>ELSE 'van '    to_char(bestuursleden.begin_datum,<br>'DD/MM')    ' tot '    to_char(bestuursleden.eind_datum,<br>'DD/MM/YYYY')<br>END AS periode<br>FROM bestuursleden, spelers<br>WHERE bestuursleden.spelersnr = spelers.spelersnr<br>AND spelers.geb_datum < '1970-08-05'<br>ORDER BY spelers.naam, bestuursleden.begin_datum |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef een overzicht van alle spelers met een bondsnummer.<br>Sorteer als volgt:<br>- Eerst moeten alle spelers uit zoetermeer getoond worden, deze sorteer je op naam. (spelers met dezelfde naam uit zoetermeer worden gesorteerd op bondsnr)<br>- Daarna de andere spelers, gesorteerd op bondsnr.<br><br>Tip: je kan CASE ook gebruiken in de ORDER BY component (zet dan '0')      | SELECT spelersnr, naam, plaats, bondsnr<br>FROM spelers<br>WHERE bondsnr IS NOT NULL<br>ORDER BY<br>CASE<br>WHEN plaats = 'Zoetermeer' THEN '0'<br>ELSE bondsnr<br>END, naam, bondsnr;  |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef een lijst van alle mogelijke dubbelcombinaties tussen spelers onderling.<br>Sorteer op "eerste speler" en vervolgens op "tweede speler".   | select s.voorletters    ' '    s.naam as "eerste speler",<br>t.voorletters    ' '    t.naam as "tweede speler" from spelers<br>as s, spelers as t<br>where s.spelersnr <> t.spelersnr<br>order by "eerste speler", "tweede speler"  |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Selecteer de naam van de spelers die Voorzitter geweest zijn, en de periode in 1 kolom. Sorteer van meest recent naar minst recent. (dus niet de huidige voorzitter(s))   | SELECT s.naam as voorzitter, begin_datum    ' '   <br>eind_datum as periode<br>FROM SPELERS as s, BESTUURSLEDEN as b<br>WHERE<br>s.spelersnr = b.spelersnr<br>AND functie = 'Voorzitter'<br>AND eind_datum IS NOT NULL<br>ORDER BY periode desc   |   |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef van elke speler die enkel wedstrijden gewonnen heeft voor team nr 1 en voor wie in totaal meer dan 100 euro aan boete betaald is, het spelersnummer, zijn naam, woonplaats en het totale boetebedrag.<br>Dit resultaat moet aflopend geordend worden op het totale boetebedrag. Sorteer van voor naar achter.  | SELECT s.spelersnr, s.naam, s.plaats, sum(b.bedrag)<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>WHERE s.spelersnr IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden w<br>WHERE gewonnen > verloren<br>AND teamnr=1)<br>GROUP BY s.spelersnr, s.naam, s.plaats<br>HAVING sum(b.bedrag)>100<br>order by 1,2,3,4  |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef van elke speler die minstens 1 wedstrijd gewonnen heeft voor team nr 1 en voor wie in totaal meer dan 100 euro aan boete betaald is, het spelersnummer, zijn naam, woonplaats en het totale boetebedrag. Sorteer van voor naar achter.   | SELECT s.spelersnr, s.naam, s.plaats, sum(b.bedrag)<br>FROM spelers s INNER JOIN boetes b on<br>(b.spelersnr=s.spelersnr)<br>WHERE s.spelersnr IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden w<br>WHERE gewonnen > verloren<br>AND teamnr=1)<br>GROUP BY s.spelersnr, s.naam, s.plaats<br>HAVING sum(b.bedrag)>100<br>order by 1,2,3,4;   |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Selecteer de namen van voorzitters en de periode (= begindatum streepje einddatum) waarin ze voorzitter waren (of nog steeds zijn) en op welke datum ze beboet zijn geweest.<br>Gebruik   .   | select s.naam, CASE WHEN be.eind_datum IS NOT<br>NULL THEN to_char(be.begin_datum, 'DD/MM/YYYY')    '<br>' - '    to_char(be.eind_datum, 'DD/MM/YYYY') ELSE<br>to_char(be.begin_datum, 'DD/MM/YYYY')    ' ' - ' END as<br>periode, bo.datum<br>from bestuursleden as be, boetes as bo, spelers as s<br>where be.spelersnr = bo.spelersnr<br>and functie = 'Voorzitter'<br>and s.spelersnr = be.spelersnr  |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Sorteer de teams in functie van het aantal verloren wedstrijden (oplopend). Als het aantal verloren wedstrijden gelijk is, sorteer je op het totaal aantal gewonnen sets (aflopend) en op divisienaam (oplopend).<br>Een wedstrijd die verloren werd met 3 - 0 telt niet mee in deze output.<br>Geef het aantal wedstrijden dat voldoet aan bovenstaande voorwaarde mee in de output. | SELECT teams.divisie, COUNT(*) AS aantal<br>FROM wedstrijden, teams<br>WHERE wedstrijden.teamnr = teams.teamnr<br>AND wedstrijden.verloren > wedstrijden.gewonnen<br>AND NOT (wedstrijden.verloren = 3 AND<br>wedstrijden.gewonnen = 0)<br>GROUP BY teams.divisie<br>ORDER BY aantal, SUM(wedstrijden.gewonnen) DESC,<br>teams.divisie;   |  |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef voor alle huidige bestuurleden hun functie en de lijst van boetes die voor hen werd betaald.<br>Omdat je dit wil vergelijken met de boetebedragen die betaald werden voor spelers die niet in het bestuur zitten, wil je deze boetebedragen ook opnemen in de tweede kolom van je resultaat. Sorteer je antwoord eerst op functie en daarna op het boetebedrag.                  | SELECT functie, bedrag<br>FROM boetes BT FULL OUTER JOIN bestuursleden B<br>ON (BT.spelersnr = B.spelersnr<br>AND eind_datum is null)<br>ORDER BY 1,2;  |  |
|      |                                  |   | SELECT spelersnr AS veld1, naam AS veld2, jaartoe AS<br>veld3   |   |











|      |                                  |   |  |   |
|------|----------------------------------|---|--|---|
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef een overzicht van alle spelers, gevolgd door alle bestuursleden, gesorteerd op jaar van toetreding of beginjaar van hun functie en vervolgens op spelersnr.<br>Geen dubbels tonen.   | FROM spelers<br>UNION<br>SELECT spelersnr AS veld1, functie AS veld2, EXTRACT (YEAR FROM begin_datum) AS veld3<br>FROM bestuursleden<br>ORDER BY veld3, veld1  |     |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef een lijst van alle mogelijke dubbelcombinaties van spelers. Beperking: de verschillende spelers mogen geen kapitein zijn van hetzelfde team (moest dit mogelijk zijn). Orden eerst naar speler1, dan naar speler2.   | select DISTINCT s.voorletters    ' '    s. naam AS "speler1",<br>t.voorletters    ' '    t. naam AS "speler2"<br>from spelers as s, spelers as t, teams as te, teams as tt<br>where s.spelersnr <> t.spelersnr AND<br>NOT (s.spelersnr = te.spelersnr AND t.spelersnr =<br>tt.spelersnr AND te.teamnr = tt.teamnr)<br>ORDER BY 1,2   |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef het gemiddeld boetebedrag per speler, afgerond op twee cijfers na de komma. spelers zonder boete moeten bovenaan staan met 'geen boetes'<br>Sorteer daarna op spelersnaam en vervolgens op gemiddeld boetebedrag.<br>Gebruik als datatype van de 2de kolom varchar(8) voor spelers die wel boetes hebben.                                      | SELECT spelers.naam,<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 'geen boetes'<br>ELSE CAST(ROUND(AVG(boetes.bedrag), 2) AS<br>varchar(8))<br>END<br>AS gemiddeld<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr =<br>boetes.spelersnr<br>GROUP BY spelers.spelersnr, spelers.naam<br>ORDER BY<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 0<br>ELSE 1<br>END, spelers.naam, AVG(boetes.bedrag);   |    |
| 2023 | Databanken: DB2<br>Examenvraag 1 | Geef per team de leeftijd van de aanvoerder (tip: postgresql heeft een AGE() functie) en het aantal verschillende spelers dat voor dit team gespeeld heeft. Alleen teams waarvoor wedstrijden zijn gespeeld en die een aanvoerder hebben, moeten vermeld worden. Sorteer op leeftijd en daarna op aantal verschillende spelers en daarna op teamnr. | SELECT teams.teamnr, CAST(EXTRACT(year from<br>AGE(NOW(), spelers.geb_datum)) AS varchar(2))    ' jaar'<br>AS leeftijd, COUNT(DISTINCT wedstrijden.spelersnr) AS<br>aantalspelers<br>FROM teams<br>INNER JOIN spelers ON teams.spelersnr =<br>spelers.spelersnr<br>INNER JOIN wedstrijden ON teams.teamnr =<br>wedstrijden.teamnr<br>GROUP BY teams.teamnr, spelers.geb_datum<br>ORDER BY leeftijd, aantalspelers, teamnr; |    |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef het hemellichaam dat het laatst bezocht is.<br>Gebruik hiervoor de laatste vertrekdatum van de reis en laatste volgnummer van bezoek. Tip: gebruik hiervoor een rij-subquery.<br>Gebruik geen limit of top.  | SELECT objectnaam<br>FROM bezoeken<br>WHERE (reisnr, volgnr) = (<br>SELECT bezoeken.reisnr, MAX(bezoeken.volgnr)<br>FROM bezoeken<br>INNER JOIN reizen laatstereis USING (reisnr)<br>WHERE laatstereis.vertrekdatum = (<br>SELECT MAX(vertrekdatum)<br>FROM reizen<br>)<br>)<br>GROUP BY bezoeken.reisnr<br>)  |  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef de planeten en het aantal verschillende personen die hen bezocht hebben (met of zonder verblijf).<br>Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam, COUNT(DISTINCT<br>deelnames.klantnr) AS aantalbezoekers<br>FROM hemelobjecten<br>LEFT OUTER JOIN bezoeken ON<br>hemelobjecten.objectnaam = bezoeken.objectnaam<br>LEFT OUTER JOIN deelnames ON bezoeken.reisnr =<br>deelnames.reisnr<br>WHERE hemelobjecten.satellietvan = 'Zon'<br>GROUP BY hemelobjecten.objectnaam<br>ORDER BY hemelobjecten.objectnaam;                               |  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef alle planeten (gesorteerd op afstand van de zon) en het aantal verschillende reizen die deze planeet bezocht hebben.   | SELECT hemelobjecten.objectnaam, COUNT(DISTINCT<br>bezoeken.reisnr) AS aantalbezoeken<br>FROM hemelobjecten<br>LEFT OUTER JOIN bezoeken ON<br>hemelobjecten.objectnaam = bezoeken.objectnaam<br>WHERE satellietvan = 'Zon'<br>GROUP BY hemelobjecten.afstand,<br>hemelobjecten.objectnaam<br>ORDER BY afstand;   |  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Sorteer de klanten aflopend op gemiddelde kostprijs per bezoek (totaalprijs van alle reizen per klant/totaal aantal dagen verblijf op een hemelobject), op twee cijfers na de komma afgerond, daarna op klantnr.  | SELECT klanten.klantnr,<br>ROUND(SUM(reizen.prijs)/SUM(bezoeken.verblijfsduur),2)<br>AS gemiddeldeprijs<br>FROM klanten<br>INNER JOIN deelnames ON klanten.klantnr =<br>deelnames.klantnr<br>INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br>INNER JOIN bezoeken ON reizen.reisnr =<br>bezoeken.reisnr<br>GROUP BY klanten.klantnr<br>ORDER BY gemiddeldeprijs DESC, klantnr;                                     |  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef alle niet-bezochte hemelobjecten, buiten het grootste hemellichaam.<br>Sorteer op diameter en objectnaam.  | SELECT objectnaam, afstand, diameter<br>FROM hemelobjecten<br><br>WHERE diameter < ANY (<br>SELECT diameter<br>FROM hemelobjecten rest<br>WHERE rest.objectnaam <> hemelobjecten.objectnaam<br>)<br><br>AND NOT EXISTS (<br>SELECT *<br>FROM bezoeken<br>WHERE bezoeken.objectnaam =<br>hemelobjecten.objectnaam<br>)<br>ORDER BY diameter, objectnaam;  |  |
|      |                                  |   | SELECT hemelobjecten.objectnaam<br>FROM hemelobjecten  |   |

|      |                                  |   |   |   |
|------|----------------------------------|---|---|---|
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef de hemelobjecten die nog nooit bezocht zijn (een bezoek heeft een verblijfsduur van minstens één dag).   | LEFT OUTER JOIN bezoeken ON hemelobjecten.objectnaam = bezoeken.objectnaam AND bezoeken.verblijfsduur > 0<br>WHERE bezoeken.objectnaam IS NULL<br>ORDER BY hemelobjecten.objectnaam;  |     |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef per klant het totaal aantal reizen waaraan deze klant zal deelnemen en het langste bezoek dat deze klant zal maken aan een hemelobject, over alle reizen heen. Klanten zonder reizen of zonder bezoeken moeten ook voorkomen in het overzicht. Sorteer op klantnr.   | SELECT klanten.klantnr, COUNT(DISTINCT reizen.reisnr) AS aantal, MAX(bezoeken.verblijfsduur) AS langstebezoek<br>FROM klanten<br>LEFT OUTER JOIN deelnames ON klanten.klantnr = deelnames.klantnr<br>LEFT OUTER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br>LEFT OUTER JOIN bezoeken ON deelnames.reisnr = bezoeken.reisnr<br>GROUP BY klanten.klantnr<br>ORDER BY klanten.klantnr;  |    |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef de diameter van de grootste, niet bezochte maan (satelliet van een planeet).   | SELECT MAX(manen.diameter) AS grootstemaan<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten manen ON planeten.objectnaam = manen.satellietvan AND planeten.satellietvan = 'Zon'<br>LEFT OUTER JOIN bezoeken ON manen.objectnaam = bezoeken.objectnaam<br>WHERE bezoeken.reisnr IS NULL;   |    |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef voor elk hemelobject de minimale en maximale gemiddelde afstand tot zijn zon (de centrale ster in een sterrenstelsel) als je weet dat de kolom 'afstand' de gemiddelde afstand bevat tot het hemelobject waar rond ze draaien. Met de grootte van het hemelobject hoeft u geen rekening te houden. Sorteer op minimale afstand en op objectnaam. | SELECT hemelobjecten.objectnaam, CASE WHEN hemelobjecten.afstand IS NULL THEN 0 WHEN draaitrond.afstand IS NULL THEN hemelobjecten.afstand ELSE draaitrond.afstand + hemelobjecten.afstand END AS maximale_afstand, CASE WHEN hemelobjecten.afstand IS NULL THEN 0 WHEN draaitrond.afstand IS NULL THEN hemelobjecten.afstand ELSE draaitrond.afstand - hemelobjecten.afstand END AS minimale_afstand<br>FROM hemelobjecten<br>LEFT OUTER JOIN hemelobjecten draaitrond ON hemelobjecten.satellietvan = draaitrond.objectnaam<br>ORDER BY minimale_afstand, objectnaam; |    |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef de planeet (draait dus rond de zon) met de meeste satellieten. Sorteer op objectnaam.  | SELECT planeten.objectnaam<br>FROM hemelobjecten planeten<br>LEFT OUTER JOIN hemelobjecten satellieten ON planeten.objectnaam = satellieten.satellietvan<br>WHERE planeten.satellietvan = 'Zon'<br>GROUP BY planeten.objectnaam<br>HAVING COUNT(*) = (SELECT MAX(aantalsatellieten) FROM (SELECT COUNT(*) AS aantalsatellieten FROM hemelobjecten planeten LEFT OUTER JOIN hemelobjecten satellieten ON planeten.objectnaam = satellieten.satellietvan WHERE planeten.satellietvan = 'Zon' GROUP BY planeten.objectnaam ) AS aantallen)<br>ORDER BY planeten.objectnaam |  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef de diameter van het grootste hemellichaam dat bezocht is op de vroegste reis waar klantnr 126 niet op meegegaan is.  | SELECT MAX(diameter) AS grootste<br>FROM hemelobjecten<br>INNER JOIN bezoeken USING (objectnaam)<br>INNER JOIN reizen USING (reisnr)<br>WHERE vertrekdatum = (SELECT MIN(vertrekdatum) FROM reizen<br>INNER JOIN bezoeken USING (reisnr)<br>WHERE reisnr NOT IN (SELECT reisnr FROM deelnames WHERE klantnr = 126)<br>)   |  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef de volledige frequentietabel voor de diameters van de hemelobjecten (frequentie: hoeveel objecten zijn er met de gegeven diameter, cumulatieve Frequentie, relatieve frequentie, Relatieve cumulatieve frequentie). Let op de datatypes en de precisie, gebruik CAST, rond niet af. Sorteer op diameter.   | SELECT b.diameter AS diameter, f, count(*) as "cf", CAST(f*100.00/tot AS NUMERIC(5,2))AS rf, CAST(100.00*count(*)/tot AS NUMERIC(5,2)) as "crf"<br>FROM (SELECT diameter, count(*) AS f FROM hemelobjecten GROUP BY diameter) AS b<br>INNER JOIN hemelobjecten bo ON (b.diameter >= bo.diameter),<br>(SELECT count(*) as tot FROM hemelobjecten) AS boe<br>GROUP BY b.diameter, f, tot<br>ORDER BY 1;   |  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef het op één na kleinste hemellichaam. Je kan dit vinden door handig gebruik te maken van expliciete joins en een doorsnedevoorwaarde. Tip: probeer eerst een lijst te krijgen van alle hemelobjecten en het aantal hemellichaam dat kleiner is dan dat hemelobject.   | SELECT zelf.objectnaam, count(kleiner.objectnaam) AS aantalkleiner<br>FROM hemelobjecten zelf<br>LEFT OUTER JOIN hemelobjecten kleiner ON zelf.diameter > kleiner.diameter<br>GROUP BY zelf.objectnaam<br>HAVING count(kleiner.objectnaam) = 1  |  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Maak een lijst met die mensen die meer dan 2 maal een reis ondernomen hebben waarin men geen enkele satelliet van Jupiter bezoekt !. Sorteer van voor naar achter.  | SELECT k.klantnr, (k. naam    k.vnaam) AS "Volledige naam", count(d.reisnr) AS "Aantal Ondernomen Reizen"<br>FROM klanten k JOIN deelnames d USING (k.klantnr)<br>WHERE d.reisnr NOT IN (SELECT b.reisnr FROM bezoeken b NATURAL INNER JOIN hemelobjecten o WHERE o.satellietvan = 'Jupiter')<br>GROUP BY k.klantnr, k. naam    k.vnaam   |  |

|      |                                     |  |  |   |
|------|-------------------------------------|--|--|---|
|      |                                     |  | HAVING count(d.reisnr)>2<br>order by 1,2,3   |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Hoeveel kilometers heeft iedereen in totaal gevlogen tot nu toe en hoeveel hebben ze hier in totaal voor betaald. Vermits we de posities van de planeten niet kennen, mag je de afstanden van de hemelobjecten direct gebruiken. Geef het totaal gependeerde bedrag, de afgelegde kilometers, de prijs per kilometer en datum van hun laatste vlucht van al hun persoonlijke reizen. In het geval dat iemand niet op reis is geweest of geen kilometers gedaan heeft, toon je de boodschap 'veel geld voor niks of niet op reis geweest' in de kolom prijs_per_kilometer.<br>Sorteer van voor naar achter. | SELECT k.klantnr, k. naam    ' '    k.vnaam AS naam, tot_bedrag, tot_afstand, CASE WHEN tot_afstand <> 0 THEN CAST(tot_bedrag/tot_afstand AS varchar) ELSE 'veel geld voor niks of niet op reis geweest' END AS prijs_per_kilometer, laatste_reis_datum FROM klanten k NATURAL LEFT OUTER JOIN ((SELECT klantnr, sum(prijs) AS tot_bedrag, max(vertrekdatum) AS laatste_reis_datum FROM deelnames NATURAL LEFT OUTER JOIN reizen GROUP BY klantnr) AS dr NATURAL LEFT OUTER JOIN (SELECT klantnr, sum(afstand) AS tot_afstand FROM (deelnames NATURAL INNER JOIN bezoeken) NATURAL INNER JOIN hemelobjecten GROUP BY klantnr) AS bh) ORDER BY 1,2,3,4,5,6; |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Maak een lijst van klanten die meer dan 2 keer een reis gemaakt hebben waarbij er geen bezoek was aan Jupiter.   | SELECT k.klantnr, (k. naam    ' '    k.vnaam) AS klantnaam, COUNT(d.reisnr) AS aantalreizen FROM klanten k INNER JOIN deelnames d USING (k.klantnr) WHERE d.reisnr NOT IN (SELECT b.reisnr FROM bezoeken b INNER JOIN hemelobjecten o ON b.objectnaam = o.objectnaam WHERE o.objectnaam = 'Jupiter') GROUP BY k.klantnr, k. naam, k.vnaam HAVING count(d.reisnr) > 2;  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef de klantnr voor de klant met het meeste bezoeken aan de maan. Geef ook het aantal bezoeken.<br>Gebruik geen limit of top.   | SELECT d.klantnr, count(b.objectnaam) FROM deelnames d INNER JOIN reizen r ON d.reisnr = r.reisnr INNER JOIN bezoeken b ON r.reisnr = b.reisnr WHERE b.objectnaam = 'Maan' GROUP BY d.klantnr HAVING COUNT(b.objectnaam) = (SELECT MAX(c) FROM (SELECT d.klantnr, COUNT(b.objectnaam) AS c FROM deelnames d INNER JOIN bezoeken b ON d.reisnr = b.reisnr WHERE b.objectnaam = 'Maan' GROUP BY d.klantnr ) AS temp );   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Maak een lijst met klantgegevens van de personen die nog nooit op Phobos op bezoek geweest zijn. Maak expliciet gebruik van de 'uitgezonderd' set operator. (Is dit de meest efficiënte oplossing?)<br>Sorteer van voor naar achter.   | SELECT k.klantnr, k. naam, k.vnaam FROM klanten k EXCEPT SELECT k.klantnr, k. naam, k.vnaam FROM ((klanten k INNER JOIN deelnames d using (k.klantnr)) INNER JOIN reizen r using(reisnr)) INNER JOIN bezoeken b using(reisnr) WHERE b.objectnaam = 'Phobos' ORDER BY 1,2,3;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Maak een lijst van de mensen die Mars wel bezocht hebben maar lo nog niet. Sorteer van voor naar achter.   | SELECT k.klantnr, k. naam FROM ((klanten k INNER JOIN deelnames d using (k.klantnr)) INNER JOIN reizen r using(reisnr)) INNER JOIN bezoeken b using(reisnr) WHERE b.objectnaam = 'Mars' EXCEPT SELECT k.klantnr, k. naam FROM ((klanten k INNER JOIN deelnames d using (k.klantnr)) INNER JOIN reizen r using(reisnr)) INNER JOIN bezoeken b using(reisnr) WHERE b.objectnaam = 'lo' order by 1,2  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef de klant die het meest op de maan is geweest (+het aantal).<br>Sorteer van voor naar achter.  | SELECT d.klantnr, count(b.objectnaam) FROM (deelnames d INNER JOIN reizen r USING (reisnr)) INNER JOIN bezoeken b USING (reisnr) WHERE b.objectnaam = 'Maan' GROUP BY d.klantnr HAVING count(b.objectnaam) = (SELECT max(c) FROM (SELECT d.klantnr, count(b.objectnaam) AS c FROM (deelnames d INNER JOIN reizen r USING (reisnr)) INNER JOIN bezoeken b USING (reisnr) WHERE b.objectnaam = 'Maan' GROUP BY d.klantnr) AS temp) order by 1,2  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef de diameter van de grootste, niet bezochte maan (satelliet van een planeet).  | SELECT MAX(manen.diameter) AS grootstemaan FROM hemelobjecten planeten INNER JOIN hemelobjecten manen ON planeten.objectnaam = manen.satellietvan AND planeten.satellietvan = 'Zon' LEFT OUTER JOIN bezoeken ON manen.objectnaam = bezoeken.objectnaam WHERE bezoeken.reisnr IS NULL;  |  |
| 2023 | Databanken:<br>DB2                  | Welke planeten met meer dan 7 manen worden bezocht (met of zonder verblijf)?<br>Sorteer aflopend op basis van het aantal manen.  | SELECT planeten.objectnaam FROM hemelobjecten planeten INNER JOIN hemelobjecten satellieten ON satellieten.satellietvan = planeten.objectnaam INNER JOIN bezoeken ON planeten.objectnaam = bezoeken.objectnaam   |  |











|      |                                     |   |  |   |
|------|-------------------------------------|---|--|---|
|      | Examenvraag 2                       | <p>Let erop dat je planeten die meerdere keren bezocht worden, niet dubbel telt.</p>  | <p>WHERE planeten.satellietvan = 'Zon'<br/>                     GROUP BY planeten.objectnaam<br/>                     HAVING COUNT(DISTINCT satellieten.objectnaam) &gt; 7<br/>                     ORDER BY COUNT(DISTINCT satellieten.objectnaam)<br/>                     DESC;</p>   |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | <p>Geef de naam, diameter en het langste verblijf op van alle hemelobjecten met minder dan 5 hemelobjecten die rond dit object draaien. (ze moeten dus bezocht zijn)<br/>                     Sorteer op objectnaam.</p>  | <p>SELECT hemelobjecten.objectnaam,<br/>                     hemelobjecten.diameter, MAX(bezoeken.verblijfsduur) AS<br/>                     maximale_verblijf<br/>                     FROM bezoeken<br/>                     INNER JOIN hemelobjecten ON bezoeken.objectnaam =<br/>                     hemelobjecten.objectnaam<br/>                     LEFT OUTER JOIN hemelobjecten manen ON<br/>                     hemelobjecten.objectnaam = manen.satellietvan<br/>                     GROUP BY hemelobjecten.objectnaam,<br/>                     hemelobjecten.diameter<br/>                     HAVING COUNT(DISTINCT manen.objectnaam) &lt; 5<br/>                     ORDER BY objectnaam;</p>  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | <p>Geef voor elk hemelobject de minimale en maximale gemiddelde afstand tot zijn zon (de centrale ster in een sterrenstelsel) als je weet dat de kolom 'afstand' de gemiddelde afstand bevat tot het hemelobject waar rond ze draaien. Met de grootte van het hemelobject hoeft je geen rekening te houden.<br/>                     Sorteer op minimale afstand en op objectnaam.<br/>                     Tip: bekijk de functie COALESCE om met null-waardes rekening te houden.<br/>                     Opm: als 1 van beide afstanden null is, dan veronderstellen we dat deze 0 is.</p>  | <p>SELECT<br/>                     hemelobjecten.objectnaam,<br/>                     coalesce(hemelobjecten.afstand, 0) +<br/>                     coalesce(draaitrond.afstand, 0) AS maximale_afstand,<br/>                     abs(coalesce(draaitrond.afstand, 0) -<br/>                     coalesce(hemelobjecten.afstand, 0)) AS minimale_afstand<br/>                     FROM hemelobjecten<br/>                     LEFT OUTER JOIN hemelobjecten draaitrond ON<br/>                     draaitrond.objectnaam = hemelobjecten.satellietvan<br/>                     ORDER BY minimale_afstand, objectnaam;</p>  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | <p>Geef voor elke reis het aantal klanten waarvan de naam niet met een 'G' begint en waarvan de periode van de geboortedatum van de klant tot de vertrekdatum van de reis overlapt met de huidige datum en 50 jaar verder (gebruik hiervoor de gepaste operator: OVERLAPS).<br/>                     Indien er op de reis hemelobjecten worden bezocht waarvan de tweede letter van het hemelobject voorkomt in de naam van het hemelobject waarvan dit bezocht hemelobject een satelliet is, dan wordt deze reis genegeerd.<br/>                     Sorteer op reisnr, en daarna op count</p> <p>(tip: conditie in outer join is niet hetzelfde als conditie in de where; + interval)</p> | <p>SELECT r.reisnr, count(k.klantnr)<br/>                     FROM (REIZEN r NATURAL LEFT OUTER JOIN<br/>                     deelnames)<br/>                     LEFT OUTER JOIN klanten k ON (k.klantnr =<br/>                     deelnames.klantnr)<br/>                     AND upper(naam) NOT LIKE 'G%'<br/>                     AND (geboortedatum, vertrekdatum) OVERLAPS<br/>                     (CURRENT_DATE, CURRENT_DATE+interval '50 year')<br/>                     WHERE r.reisnr NOT IN<br/>                     (SELECT reisnr<br/>                     FROM bezoeken NATURAL INNER JOIN hemelobjecten<br/>                     WHERE lower(satellietvan) LIKE '%'   <br/>                     lower(substring(objectnaam FROM 2 FOR 1))    '%')<br/>                     GROUP BY r.reisnr<br/>                     ORDER BY 1,2;</p> |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | <p>Geef per klant het totaal aantal dagen dat deze klant in totaal op een planeet zal verblijven.<br/>                     Sorteer op klantnr.</p>  | <p>SELECT klanten.klantnr, SUM(bezoeken.verblijfsduur) AS<br/>                     totaleverblijfsduur<br/>                     FROM klanten<br/>                     INNER JOIN deelnames ON klanten.klantnr =<br/>                     deelnames.klantnr<br/>                     INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br/>                     INNER JOIN bezoeken ON reizen.reisnr =<br/>                     bezoeken.reisnr<br/>                     INNER JOIN hemelobjecten P ON bezoeken.objectnaam<br/>                     = P.objectnaam inner join hemelobjecten S ON<br/>                     (P.satellietvan = S.objectnaam AND S.satellietvan IS<br/>                     NULL)<br/>                     GROUP BY klanten.klantnr<br/>                     ORDER BY klantnr;</p>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | <p>Reken per planeet uit wat de gemiddelde afstand van zijn manen is.</p>   | <p>SELECT planeet.objectnaam as planeet,<br/>                     AVG(maan.afstand) as afstand<br/>                     FROM hemelobjecten ster<br/>                     INNER JOIN hemelobjecten planeet ON ster.satellietvan<br/>                     IS NULL AND planeet.satellietvan = ster.objectnaam<br/>                     INNER JOIN hemelobjecten maan ON maan.satellietvan =<br/>                     planeet.objectnaam<br/>                     GROUP BY planeet.objectnaam<br/>                     ORDER BY afstand</p>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | <p>Vind per planeet die zijn maan die het verst ervan verwijderd is. Geef planeet, maan en afstand en sorteer stijgend op afstand.</p>  | <p>SELECT planeet.objectnaam as planeet, M.objectnaam as<br/>                     maan, M.afstand as afstand<br/>                     FROM hemelobjecten ster<br/>                     INNER JOIN hemelobjecten planeet ON ster.satellietvan<br/>                     IS NULL AND planeet.satellietvan = ster.objectnaam<br/>                     INNER JOIN hemelobjecten maan ON maan.satellietvan =<br/>                     planeet.objectnaam<br/>                     INNER JOIN hemelobjecten M on M.satellietvan =<br/>                     planeet.objectnaam<br/>                     GROUP BY planeet.objectnaam, M.objectnaam<br/>                     HAVING M.afstand = MAX(maan.afstand)<br/>                     ORDER BY afstand</p>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | <p>Geef een lijst van alle planeten in het heelal die minstens 1 satelliet hebben met een diameter groter dan 2000km (je weet dus niet hoeveel satellieten, kan ook bv 100 zijn). Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer op planeetnaam.<br/>                     Gebruik to_char .. 999 om te casten voor de count.</p>   | <p>SELECT planeet.objectnaam, CASE WHEN<br/>                     count(sat.objectnaam) = 0 THEN 'Geen' ELSE<br/>                     to_char(count(sat.objectnaam), '999') END AS<br/>                     "interessante satellieten"<br/>                     FROM hemelobjecten planeet INNER JOIN hemelobjecten<br/>                     ster ON (planeet.satellietvan = ster.objectnaam AND<br/>                     ster.satellietvan is null)<br/>                     LEFT OUTER JOIN hemelobjecten sat ON<br/>                     (sat.satellietvan = planeet.objectnaam AND sat.diameter &gt;<br/>                     2000)<br/>                     GROUP BY planeet.objectnaam<br/>                     ORDER BY planeet.objectnaam</p>   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | <p>Geef een lijst van alle planeten die minstens 1 satelliet hebben wiens naam een kleine letter i bevat. Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer op planeetnaam.<br/>                     (Gebruikt to_char .. '999') voor omzetting telling)</p>  | <p>SELECT planeet.objectnaam, CASE WHEN<br/>                     count(sat.objectnaam) = 0 THEN 'Geen' ELSE<br/>                     to_char(count(sat.objectnaam), '999') END AS "satellieten<br/>                     met een letter i"<br/>                     FROM hemelobjecten planeet INNER JOIN hemelobjecten<br/>                     ster ON (planeet.satellietvan = ster.objectnaam AND<br/>                     ster.satellietvan is null)<br/>                     LEFT OUTER JOIN hemelobjecten sat ON<br/>                     (sat.satellietvan = planeet.objectnaam AND<br/>                     sat.objectnaam LIKE '%i%')<br/>                     GROUP BY planeet.objectnaam<br/>                     ORDER BY planeet.objectnaam</p>  |  |















|      |                                     |  |   |   |
|------|-------------------------------------|--|---|---|
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef een lijst van alle planeten die minstens 1 satelliet hebben op een afstand groter dan 500km. Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer omgekeerd alfabetisch op planeetnaam.<br>(Gebruikt to_char .. '999') voor omzetting telling)   | <pre>SELECT planeet.objectnaam, CASE WHEN count(sat.objectnaam) = 0 THEN 'Geen' ELSE to_char(count(sat.objectnaam), '999') END AS "ver verwijderde satellieten" FROM hemelobjecten planeet INNER JOIN hemelobjecten ster ON (planeet.satellietvan = ster.objectnaam AND ster.satellietvan is null) LEFT OUTER JOIN hemelobjecten sat ON (sat.satellietvan = planeet.objectnaam AND sat.afstand &gt; 500) GROUP BY planeet.objectnaam ORDER BY planeet.objectnaam desc</pre>         |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Welke planeten met meer dan 7 manen worden bezocht (met of zonder verblijf)?<br>Sorteer aflopend op basis van het aantal manen.<br>Let erop dat je planeten die meerdere keren bezocht worden, niet dubbel telt.   | <pre>SELECT planeten.objectnaam FROM hemelobjecten planeten INNER JOIN hemelobjecten satellieten ON satellieten.satellietvan = planeten.objectnaam INNER JOIN hemelobjecten sterren ON planeten.satellietvan = sterren.objectnaam INNER JOIN bezoeken ON planeten.objectnaam = bezoeken.objectnaam WHERE sterren.satellietvan IS NULL GROUP BY planeten.objectnaam HAVING COUNT(DISTINCT satellieten.objectnaam) &gt; 7 ORDER BY COUNT(DISTINCT satellieten.objectnaam) DESC;</pre> |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef het reisnr, de prijs en vertrekdatum van de reis met de hoogste gemiddelde verblijfsduur op een hemelobject (=som van de verblijfsduur / aantal bezoeken per reis).   | <pre>SELECT reizen.reisnr, prijs, vertrekdatum FROM reizen INNER JOIN bezoeken USING (reisnr) GROUP BY reizen.reisnr, prijs, vertrekdatum HAVING (SUM(verblijfsduur) / COUNT(objectnaam)) = ( SELECT MAX(gemiddelde) FROM ( SELECT SUM(verblijfsduur) / COUNT(objectnaam) as gemiddelde FROM bezoeken GROUP BY reisnr ) AS reisgemiddelden )</pre>  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef de naam, diameter en het langste verblijf op alle hemelobjecten met minder dan 5 hemelobjecten die rond dit object draaien. Geef enkel hemelobjecten die bezocht zijn of waar gepasseerd wordt.<br>Sorteer op objectnaam.   | <pre>SELECT hemelobjecten.objectnaam, hemelobjecten.diameter, MAX(bezoeken.verblijfsduur) AS maximale_verblijf FROM bezoeken INNER JOIN hemelobjecten ON bezoeken.objectnaam = hemelobjecten.objectnaam LEFT OUTER JOIN hemelobjecten manen ON hemelobjecten.objectnaam = manen.satellietvan GROUP BY hemelobjecten.objectnaam, hemelobjecten.diameter HAVING COUNT(DISTINCT manen.objectnaam) &lt; 5 ORDER BY objectnaam;</pre>  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef de gemiddelde afstanden tot hun planeet van alle satellieten van Saturnus, afgerond op 2 getallen na de komma   | <pre>SELECT round(AVG(S.afstand), 2) as "Gemiddelde afstand" FROM hemelobjecten P, hemelobjecten S WHERE P.objectnaam = 'Saturnus' AND S.satellietvan = P.objectnaam</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef voor elke een klant een overzicht aan uitgaven. Hoe? Geef voor elke klant een cumulatief overzicht van prijs van de reizen waar hij aan deelgenomen heeft. De volgorde, waarin er wordt cumulatief wordt opgeteld, wordt bepaald door de vertrekdatum van de reis. Sorteer van voor naar achter.<br>Tip: vergelijk deze uitvoer met de uitvoer van een query die een gesorteerd overzicht geeft van de klanten en hun deelnames aan reizen.   | <pre>select klantnr, reisnr, sum(prijs) OVER (partition by klantnr order by vertrekdatum) from reizen natural join deelnames order by 1, 2, 3;</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef enkel de 2 voorlaatste reizen terug. De positie van reizen wordt bepaald door het reisnummer.<br>Ter vergelijking, als je de getallen 1 t/m 10 neemt, dan is dit 8 en 9. Sorteer van voor naar achter.<br>Gebruik enkel ISO sql.  | <pre>SELECT * from (select * from reizen order by reisnr desc offset 1 fetch first 2 rows only) as t order by 1,2,3,4;</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef het aantal unieke satellieten van planeten binnen ons sterrenstelsel  | <pre>SELECT count(DISTINCT S.objectnaam) as aantal_satellieten FROM hemelobjecten Z, hemelobjecten P, hemelobjecten S WHERE Z.satellietvan is null AND P.satellietvan = Z.objectnaam AND S.satellietvan = P.objectnaam</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | Geef de totale som van alle afstanden tot hun planeet van alle satellieten van Saturnus  | <pre>SELECT SUM(S.afstand) as "Totale afstand" FROM hemelobjecten P, hemelobjecten S WHERE P.objectnaam = 'Saturnus' AND S.satellietvan = P.objectnaam</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag 2 | <p>Toon in xmlformaat de objectnamen, afstand en diameter voor objecten die rond Neptunus draaien. Geef als eerste kolom de commentaar maan. Sorteer van voor naar achter.<br/>Tip: XML is gebaseerd op het datatype text.</p> <p>Verwacht uitvoer:<br/>xmlcomment   xmlforest</p> <pre> ----- &lt;!--maan--&gt;   &lt;objectnaam&gt;Despina&lt;/objectnaam&gt; &lt;afstand&gt;52.500&lt;/afstand&gt;&lt;diameter&gt;180&lt;/diameter&gt; &lt;!--maan--&gt;   &lt;objectnaam&gt;Galathea&lt;/objectnaam&gt; &lt;afstand&gt;62.000&lt;/afstand&gt;&lt;diameter&gt;150&lt;/diameter&gt; &lt;!--maan--&gt;   &lt;objectnaam&gt;Larissa&lt;/objectnaam&gt; &lt;afstand&gt;73.600&lt;/afstand&gt;&lt;diameter&gt;192&lt;/diameter&gt; &lt;!--maan--&gt;   &lt;objectnaam&gt;Naiad&lt;/objectnaam&gt; &lt;afstand&gt;48.000&lt;/afstand&gt;&lt;diameter&gt;54&lt;/diameter&gt; &lt;!--maan--&gt;   &lt;objectnaam&gt;NereÄ de&lt;/objectnaam&gt; &lt;afstand&gt;5517.000&lt;/afstand&gt;&lt;diameter&gt;240&lt;/diameter&gt; &lt;!--maan--&gt;   &lt;objectnaam&gt;Proteus&lt;/objectnaam&gt; &lt;afstand&gt;117.600&lt;/afstand&gt;&lt;diameter&gt;416&lt;/diameter&gt; &lt;!--maan--&gt;   &lt;objectnaam&gt;Thalassa&lt;/objectnaam&gt;</pre> | <pre>select xmlcomment('maan'), xmlforest(objectnaam,afstand,diameter) from hemelobjecten where satellietvan = 'Neptunus' order by cast(xmlcomment('maan') as text), cast(xmlforest(objectnaam,afstand,diameter) as text);</pre>  |  |

|      |                                  |  |  |
|------|----------------------------------|--|--|
|      |                                  | <afstand>50.000</afstand><diameter>80</diameter><br><!--maan-->   <objectnaam>Triton</objectnaam><br><afstand>354.800</afstand><diameter>2705</diameter>   |  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef de deelnemers waarbij hun aantal reizen die ze ondernemen groter is dan alle hemelobjecten (die niet beginnen met de letter 'M') hun aantal keren dat ze bezocht zijn. Of anders geformuleerd: Geef de deelnemers met meer deelnames dan het grootste aantal bezoeken aan een hemelobject dat niet met de letter 'M' begint (:deze deelnemer meer deelnames heeft dan de "grootste" .. = deze deelnemer heeft meer deelnames dan "alle" ..)<br>Sorteer op klantnr.  | <pre> SELECT klantnr, vnaam, naam, COUNT(*) AS aantaldeelnames FROM klanten INNER JOIN deelnames using (klantnr) GROUP BY klantnr, vnaam, naam HAVING COUNT(*) &gt; ALL ( SELECT COUNT(*) FROM bezoeken WHERE objectnaam NOT LIKE 'M%' GROUP BY objectnaam ) ORDER BY klantnr,2,3,4;  SELECT cast(r.reisnr as char(10)) AS mr_en_naam, cast(r.vertrekdatum As char(10)) as dtm_en_naam,reisduur, prijs FROM reizen r UNION SELECT * FROM hemelobjecten WHERE objectnaam like 'A%' ORDER BY 1,2,3,4;                     </pre> |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Maak een lijst met de reizen, gevolgd door de hemelobjecten wiens objectnaam met een A begint. Sorteer van voor naar achter. TIP: Gebruik CAST AS CHAR(10) voor conversies het reisnr en vertrekdatum.   | <pre> SELECT reisnr, volgnr, objectnaam, verblijfsduur, sum(verblijfsduur) OVER(partition by reisnr order by volgnr) as inc_duur, sum(verblijfsduur) over () as tot_duur FROM bezoeken ORDER by 1,2,3,4,5,6;                     </pre>  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef per reis incrementeel de totale verblijfsduur volgens de volgorde waarin de hemelobjecten bezocht worden. Geef ook de totale verblijfsduur van alle reizen om alles in perspectief te zetten. Sorteer op reisnr, volgnr, objectnaam, verblijfsduur, de volgende kolommen.   | <pre> SELECT reisnr, lag(reisnr) OVER w1 as vorig_reisnr, vertrekdatum, vertrekdatum - lag(vertrekdatum) OVER w1 as tussen_tijd, reisduur, extract(year from vertrekdatum) as jaar, sum(reisduur) OVER (partition by extract(year from vertrekdatum) order by vertrekdatum) as jaar_duur FROM reizen WINDOW w1 as (order by vertrekdatum) ORDER BY 1,2,3,4,5,6,7;                     </pre>   |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef voor elke geboortejaar van de klanten, het aantal klanten, het kleinste klantennummer en het grootste klantennummer. Geef ook het totaal aantal klanten en het kleinste en grootste klantennummer. Sorteer van voor naar achter.  | <pre> SELECT extract(year from geboortedatum), count(*),min(klantnr),max(klantnr) FROM klanten GROUP BY ROLLUP (extract(year from geboortedatum)) ORDER BY 1,2,3,4;                     </pre>   |
| 2023 | Databanken: DB2<br>Examenvraag 2 | We willen telkens het aantal reizen en de totale prijs van de volgende situaties. Voor elke maand van vertrek, voor elk tential van de reisduur, voor de combinatie van maand van vertrek en tential van de reisduur en het het totale plaatje. Sorteer van voor naar achter.  | <pre> SELECT extract(month from vertrekdatum),round(reisduur/10), count(*), sum(prijs) FROM reizen GROUP BY CUBE (extract(month from vertrekdatum),round(reisduur/10)) ORDER by 1,2,3,4;                     </pre>  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef voor de hemelobjecten een diameter groter dan 1000 het aantal hemelobjecten en de gemiddelde diameter per groep die aan het volgende voldoet. We zien per object zien hoeveel satellieten hieraan voldoen, daarnaast in combinatie met hetzelfde object en afstand per 100tal. Alsook een algemeen overzicht. Sorteer van voor naar achter.   | <pre> SELECT objectnaam, round(afstand/100),satellietvan, count(*), avg(diameter) FROM hemelobjecten WHERE diameter &gt; 1000 GROUP BY ROLLUP (satellietvan, (objectnaam,round(afstand/100)));                     </pre>  |
| 2023 | Databanken: DB2<br>Examenvraag 2 | Geef een lijst van alle hemelobjecten die meer keer bezocht gaan worden dan Jupiter. (onafhankelijk van het aantal deelnames)<br>Sorteer op objectnaam.  | <pre> SELECT hemelobjecten.objectnaam, hemelobjecten.diameter FROM hemelobjecten INNER JOIN bezoeken USING (objectnaam) GROUP BY hemelobjecten.objectnaam, hemelobjecten.diameter HAVING COUNT(*) &gt; ( SELECT COUNT(*) FROM bezoeken WHERE objectnaam = 'Jupiter' ) ORDER BY hemelobjecten.objectnaam                     </pre>   |
| 2023 | Databanken: DB2<br>Examenvraag 3 | Zet logische replicatie op tussen de server fuji.ucll.be op poort 51920 en je eigen lokale databank server op je pc. Zorg dat deze replicatie MINSTENS 1 UUR onafgebroken blijft draaien voor het examen gedaan is.<br><br>Gebruik als naam voor je REPLICATIE SLOT "SLOT_studentnr" waarbij je studentnr vervangt door jouw concrete studentnr.<br><br>DOE DIT VOOR DE DATABANK DIE LETTERLIJK DE NAAM HEEFT VAN JE STUDENTNR. DIT IS JE STRICT PERSOONLIJKE DATABANK. BV indien je STUDENT r0123456 bent, dan gaat het over DATABANK r0123456; waarbij je het SLOT SLOT_r0123456 noemt.<br><br>Test deze replicatie door te kijken of een insert op fuji bij jou op je lokale db aankomt. Indien je hiervoor code op de fuji moet uitvoeren, dan geef je deze hier in sqldropbox in als oplossing (in dat geval is het normaal dat sqldropbox klaagt, vermits het geen select statement is)<br><br>Zoals op een productiedatabank is het aantal connecties beperkt, hou dit in het achterhoofd.<br>Indien u de DATABANK 'STUK' maakt, dan is dit UW VERANTWOORDELIJKHEID. Ook voor het verdere verloop van het examen. |  |
| 2023 | Databanken: DB2                  | Schrijf 2 sql scripts.<br><br>De opgaven hiervoor staan in je persoonlijke git repository op   |  |










|               |  |   |   |
|---------------|--|---|---|
| Examenvraag 4 |  | projectwerk onder de map db2_juni, dit is ook de plaats je hem indient.   |   |
| 2023          | Databanken:<br>Exameninfo<br>Juni Augustus | <p>Lees eerst dit alvorens aan het examen te beginnen.</p> <p>Het is een examen via pc zonder algemene internettoegang.</p> <p>De maximumduur van het examen is 2 uur 30 minuten. Het examen zelf is voorzien om opgelost te kunnen worden op 1 uur 30 minuten. Er wordt geen rekening gehouden met antwoorden die te laat of in de verkeerde vorm worden ingediend.</p> <p>Enkel antwoorden die op de gevraagde manier worden ingediend tellen; er wordt geen rekening gehouden met andere vormen van indiening. Je dient dus op de juiste manier in te dienen. Bv een query vraag in sqldropbox dient via sqldropbox te worden ingediend en niet anders.</p> <p>Dit examen is persoonlijk per student, het gedrag of de behandeling omtrent het naleven van deze regels van andere studenten -die zich mogelijk niet aan deze regels houden- geven u niet het recht de hier gegeven regels te overtreden. U werkt strikt individueel Het examenreglement is van toepassing. Overtredingen van de hier en in het OER gegeven regels worden beschouwd als fraude.</p> <p>Bij de minste onduidelijkheid van deze regels brengt u onmiddellijk de beschikbare lector hiervan op de hoogte.</p> <p>Het is jouw taak om te zorgen dat benodigde hardware, software en internet werken; net als de andere vereisten. Zie <a href="https://projectwerk.ucll.be/projects/db2/wiki/Evaluatie%20voor%20het%20overzicht">https://projectwerk.ucll.be/projects/db2/wiki/Evaluatie voor het overzicht</a>. Gelieve deze ruim op voorhand door te nemen en in orde te brengen (voor de laatste les die door de lectoren zelf gegeven wordt).</p> <p>Hierbij stemt u in met deze regels en geeft u te kennen deze zonder misverstanden begrepen te hebben, dit doet u door eerst bij aanvang van het examen u naam, handtekening en de woorden 'gelezen en goedgekeurd' te onderschrijven alvorens verder te werken. Dit doe je door letterlijk "select 'gelezen en goedgekeurd'" in te geven in sqldropbox.</p>  | <p>select 'gelezen en goedgekeurd';</p>    |
| 2023          | Databanken:<br>DB2<br>Exameninfo 2         | <p>Lees eerst dit alvorens aan het examen te beginnen.</p> <p>Het is een open boek examen via pc, u mag dus verschillende papieren en elektronische bronnen gebruiken. Belangrijk is dat dit gebruik 'niet-interactief' is. Interactief gebruik is niet toegelaten en wordt beschouwd als fraude. Een voorbeeld van interactief gebruik is aan een derde -direct of indirect- hulp te vragen.</p> <p>De maximumduur van het examen is 2 uur 36 minuten. Het examen zelf is voorzien om opgelost te kunnen worden op 1 uur 30 minuten. Er wordt geen rekening gehouden met antwoorden die te laat of in de verkeerde vorm worden ingediend.</p> <p>Enkel antwoorden die op de gevraagde manier worden ingediend tellen; er wordt geen rekening gehouden met andere vormen van indiening. Je dient dus op de juiste manier in te dienen. Bv een query vraag in sqldropbox dient via sqldropbox te worden ingediend en niet anders.</p> <p>Dit examen is persoonlijk per student, het gedrag of de behandeling omtrent het naleven van deze regels van andere studenten -die zich mogelijk niet aan deze regels houden- geven u niet het recht de hier gegeven regels te overtreden. U werkt strikt individueel Het examenreglement is van toepassing. Overtredingen van de hier en in het OER gegeven regels worden beschouwd als fraude.</p> <p>Bij de minste onduidelijkheid van deze regels brengt u onmiddellijk de beschikbare lector hiervan op de hoogte.</p> <p>De algemene richtlijnen staan op: <a href="https://intranet.ucll.be/nl/student/studeren-aan-ucll/pba-de-toegestane-informatica-proximus/examen-en-evaluatie/richtlijnen-voor-studenten-bij-online-evalueren">https://intranet.ucll.be/nl/student/studeren-aan-ucll/pba-de-toegestane-informatica-proximus/examen-en-evaluatie/richtlijnen-voor-studenten-bij-online-evalueren</a> Gelieve deze op voorhand door te nemen.</p> <p>Het is jouw taak om te zorgen dat benodigde hardware, software en internet werken.</p> <p>De vak specifieke invulling staat op: <a href="https://projectwerk.ucll.be/projects/db2/wiki/Alternatieve_examinate">https://projectwerk.ucll.be/projects/db2/wiki/Alternatieve_examinate</a> Gelieve deze op voorhand door te nemen.</p> <p>Hierbij stemt u in met deze regels en geeft u te kennen deze zonder misverstanden begrepen te hebben, dit doet u door eerst bij aanvang van het examen u naam, handtekening en de woorden 'gelezen en goedgekeurd' te onderschrijven alvorens verder te werken. Dit doe je door letterlijk " select 'gelezen en goedgekeurd' " in te geven in sqldropbox.</p> | <p>select 'gelezen en goedgekeurd';</p>    |
| 2023          | Databanken:<br>DB2<br>Examenvraag A        | <p>Welke spelers hebben voor alle teams gespeeld uit de teamstabel ?<br/>(= voor welke speler bestaat er geen enkel team waar de betreffende speler nooit voor gespeeld heeft). Sorteer op spelers nummer. Gebruik de exists operator.</p>  | <pre>SELECT s.spelersnr FROM spelers s WHERE NOT EXISTS (   SELECT '1'   FROM teams t   WHERE NOT EXISTS   (     SELECT '1'     FROM wedstrijden w     WHERE w.spelersnr = s.spelersnr     AND w.teamnr = t.teamnr)) ORDER BY 1</pre>  |
| 2023          | Databanken:<br>DB2<br>Examenvraag A        | <p>Geef voor alle spelers die geen penningmeester zijn of zijn geweest alle gewonnen wedstrijden, gesorteerd op wedstrijdnummer.</p>  | <pre>SELECT spelersnr, wedstrijdnr FROM wedstrijden WHERE spelersnr NOT IN (   SELECT spelersnr   FROM bestuursleden   WHERE functie = 'Penningmeester') AND gewonnen &gt; verloren ORDER BY wedstrijdnr;</pre>                        |
|               |  | <p>Geef een lijst met de spelers die ooit bestuurslid zijn geweest (of nog steeds zijn) en niet in Den Haag of Zoetermeer wonen.</p>  | <pre>SELECT spelers.naam, COUNT(*) AS aantal</pre>  |

|      |                                     |  |  |   |
|------|-------------------------------------|--|--|---|
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Bijkomend mag deze speler maximaal 2 keer in het bestuur van de club gezeteld hebben (nu of vroeger).<br>De lijst moet aflopend gesorteerd worden op het aantal maal dat de betreffende speler in het bestuur zetelde. Mensen met hetzelfde aantal keren moeten oplopend gesorteerd worden op basis van hun spelersnr.   | FROM spelers, bestuursleden<br>WHERE bestuursleden.spelersnr = spelers.spelersnr<br>AND plaats NOT IN ('Den Haag', 'Zoetermeer')<br>GROUP BY spelers.spelersnr, spelers.naam<br>HAVING COUNT(*) <= 2<br>ORDER BY aantal DESC, spelers.spelersnr;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef voor alle bestuursleden een overzicht met hun naam, functie, de leeftijd die ze hadden op het moment van de start van hun functie en de leeftijd die ze hadden op het moment van het einde van hun functie.<br><br>ORDER BY 1,2,3,4,5<br><br>Tip: gebruik de postgresqlfuncties AGE en EXTRACT om het aantal jaar te krijgen. En CAST(...) as CHAR(2).<br><br>(Dit is een moeilijke oefening) | SELECT spelers.spelersnr, spelers.naam, bestuursleden.functie, CAST(EXTRACT(YEAR FROM AGE(begin_datum, geb_datum)) AS CHAR(2))    ' jaar' AS leeftijd_begin, CASE WHEN eind_datum IS NULL THEN 'nog bezig' ELSE CAST(EXTRACT(YEAR FROM AGE(eind_datum, geb_datum)) AS CHAR(2))    ' jaar' END AS leeftijd_einde<br>FROM spelers, bestuursleden<br>WHERE spelers.spelersnr = bestuursleden.spelersnr<br>ORDER BY 1,2,3,4,5;                                 |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Maak een lijst met de spelers (naam van de speler, voorletter en woonplaats) die ooit gespeeld hebben voor een team dat nu in de tweede divisie speelt en waarvoor geen enkele boete betaald werd voor 1 januari 1981. Sorteer van voor naar achter, oplopend. Zorg dat er geen dubbels worden getoond.  | SELECT DISTINCT s.naam, s.voorletters, s.plaats<br>FROM spelers s INNER JOIN wedstrijden w USING (spelersnr)<br>WHERE w.teamnr IN (<br>SELECT teamnr<br>FROM teams<br>WHERE divisie = 'tweede')<br>AND s.spelersnr NOT IN (<br>SELECT spelersnr<br>FROM boetes<br>WHERE datum < '1-1-1981')<br>ORDER BY 1,2,3;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef het gemiddeld aantal gewonnen en verloren sets per geboortjaar.<br>Sorteer op geboortjaar.  | SELECT EXTRACT(YEAR FROM geb_datum) AS geboortjaar, AVG(gewonnen) AS gewonnen, AVG(verloren) AS verloren<br>FROM wedstrijden, spelers<br>WHERE wedstrijden.spelersnr = spelers.spelersnr<br>GROUP BY EXTRACT(YEAR FROM geb_datum)<br>ORDER BY geboortjaar;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Je kan per speler berekenen hoeveel boetes die speler heeft gehad en wat het totaalbedrag per speler is. Pas nu deze query aan zodat per verschillend aantal boetes wordt getoond hoe vaak dit aantal boetes voorkwam.Sorteer eerst op de eerste kolom en daarna op de tweede kolom.   | SELECT a, count(a)<br>FROM (SELECT count(bedrag) AS a<br>FROM boetes<br>GROUP BY spelersnr) b<br>GROUP BY a<br>ORDER BY 1,2  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef de twee laagste bondnrs terug. (tip: dwz er zijn dus minder dan 2 bondsnr die kleiner zijn). Sorteer op bondnr. Zonder het gebruik van LIMIT.   | SELECT s.bondsnr<br>FROM spelers s<br>WHERE 2 > (<br>SELECT count(*)<br>FROM spelers sub<br>WHERE sub.bondsnr < s.bondsnr)<br>AND s.bondsnr IS NOT NULL<br>ORDER BY 1;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef spelers die in het jaar dat ze lid geworden zijn van de club reeds een boete van meer dan 50 euro gekregen hebben en de som van al deze boetes groter of gelijk is aan 100 euro.<br>Geef buiten de voorletters en de naam van de speler ook het aantal boetes die aan bovenstaande voorwaarden voldoen.<br>Sorteer op spelersnr.  | SELECT spelers.voorletters, spelers.naam, COUNT(*) AS aantalboetes<br>FROM spelers, boetes<br>WHERE spelers.spelersnr = boetes.spelersnr<br>AND boetes.bedrag > 50<br>AND EXTRACT(YEAR FROM boetes.datum) = spelers.jaartoe<br>GROUP BY spelers.spelersnr, spelers.voorletters, spelers.naam<br>HAVING SUM(bedrag) >= 100<br>ORDER BY spelers.spelersnr;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef voor de actieve bestuursleden zonder boete hun laatste gespeelde wedstrijd (die met het hoogste wedstrijdnummer). Sorteer aflopend op spelersnr.  | SELECT bestuursleden.spelersnr, MAX(wedstrijden.wedstrijdnr) AS laatstewedstrijd<br>FROM bestuursleden<br>INNER JOIN wedstrijden ON bestuursleden.spelersnr = wedstrijden.spelersnr AND bestuursleden.eind_datum IS NULL<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr = boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY bestuursleden.spelersnr<br>ORDER BY spelersnr DESC;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef per team de leeftijd van de aanvoerder (tip: postgresql heeft een AGE() functie) en het aantal verschillende spelers dat voor dit team gespeeld heeft.<br>Alleen teams waarvoor wedstrijden zijn gespeeld en die een aanvoerder hebben, moeten vermeld worden.<br>Sorteer op leeftijd en daarna op aantal verschillende spelers en daarna op teamnr.  | SELECT teams.teamnr, CAST(EXTRACT(year from AGE(NOW(), spelers.geb_datum)) AS varchar(2))    ' jaar' AS leeftijd, COUNT(DISTINCT wedstrijden.spelersnr) AS aantalspelers<br>FROM teams<br>INNER JOIN spelers ON teams.spelersnr = spelers.spelersnr<br>INNER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr<br>GROUP BY teams.teamnr, spelers.geb_datum<br>ORDER BY leeftijd, aantalspelers, teamnr;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef alle spelers die meer wedstrijden gespeeld hebben dan het aantal wedstrijden dat de huidige voorzitter heeft verloren. De huidige voorzitter komt zelf niet in de lijst voor. Gebruik geen subqueries.<br>Sorteer op spelersnr.   | SELECT spelers.spelersnr<br>FROM spelers<br>INNER JOIN wedstrijden ON spelers.spelersnr = wedstrijden.spelersnr, bestuursleden<br>INNER JOIN wedstrijden voorzitterswedstrijden ON bestuursleden.spelersnr = voorzitterswedstrijden.spelersnr<br>AND voorzitterswedstrijden.gewonnen < voorzitterswedstrijden.verloren<br>WHERE bestuursleden.eind_datum IS NULL AND bestuursleden.functie = 'Voorzitter' AND bestuursleden.spelersnr <> spelers.spelersnr |  |



|      |                                  |   |   |   |
|------|----------------------------------|---|---|---|
|      |                                  |   | GROUP BY spelers.spelersnr<br>HAVING COUNT(DISTINCT wedstrijden.wedstrijdnr) > COUNT(DISTINCT voorzitterswedstrijden.wedstrijdnr)<br>ORDER BY spelers.spelersnr;  |   |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef voor elke aanvoerder het spelersnr, de naam en het aantal boetes dat voor hem of haar betaald is en het aantal teams dat hij of zij aanvoert. Aanvoerders zonder boetes mogen niet getoond worden. Sorteer, beginnend bij de eerste kolom, eindigend bij de laatste kolom.   | SELECT s.spelersnr, s.naam, s.voorletters, aantalboetes, aantalteams<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes,<br>(SELECT spelersnr, count(*) AS aantalteams<br>FROM teams<br>GROUP BY spelersnr) AS aantal_teams,<br>spelers s<br>WHERE s.spelersnr=aantal_boetes.spelersnr<br>AND s.spelersnr=aantal_teams.spelersnr<br>ORDER BY 1,2,3,4,5  |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef het totaal aantal boetes, het totale boetebedrag, het minimum en het maximum boetebedrag dat door onze club betaald werd. Let er hierbij op dat er gehele getallen worden getoond (rond af indien nodig). Sorteer van voor naar achter, oplopend.  | SELECT count(*) AS Aantal_boetes, round(sum(bedrag)) AS Totaal_bedrag, round(min(bedrag)) as Minimum, round(max(bedrag)) as Maximum<br>FROM boetes<br>ORDER BY 1,2,3,4;   |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Je kan per speler berekenen hoeveel boetes die speler heeft gehad en wat het totaalbedrag per speler is. Pas nu deze query aan zodat per verschillend aantal boetes wordt getoond hoe vaak dit aantal boetes voorkwam. Sorteer van voor naar achter. Probeer gelijk of beter te doen dan "Sort (cost=46.39..46.89 rows=200 width=8)". | SELECT a, count(a)<br>FROM (SELECT count(bedrag) AS a<br>FROM boetes<br>GROUP BY spelersnr) b<br>GROUP BY a<br>ORDER BY 1,2   |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef voor elke speler die een wedstrijd heeft gespeeld het spelersnr en het totaal aantal boetes. Spelers die een wedstrijd gespeeld hebben, maar geen boetes hebben, moeten ook getoond worden. Sorteer op het aantal boetes en op spelersnr;  | SELECT DISTINCT w.spelersnr, aantalboetes<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes<br>RIGHT OUTER JOIN wedstrijden w USING (spelersnr)<br>ORDER BY aantalboetes   |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef voor elke aanvoerder het spelersnr, de naam en het aantal boetes dat voor hem of haar betaald is en het aantal teams dat hij of zij aanvoert. Toon enkel aanvoerders die boetes gekregen hebben. Sorteer van voor naar achter, oplopend.   | SELECT s.spelersnr, s.naam, s.voorletters, aantalboetes, aantalteams<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes,<br>(SELECT spelersnr, count(*) AS aantalteams<br>FROM teams<br>GROUP BY spelersnr) AS aantal_teams,<br>spelers s<br>WHERE s.spelersnr=aantal_boetes.spelersnr<br>AND s.spelersnr=aantal_teams.spelersnr<br>ORDER BY 1,2,3,4,5; |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef voor de spelers die bestuurslid en/of teamkapitein zijn hun naam en een oplistng van hun functienamen (huidig of verleden) en hun divisies waarvoor ze kapitein zijn. Sorteer op spelersnaam en naam. Gebruik geen OUTER JOIN of WHERE.  | SELECT distinct spelers.naam as spelersnaam, rommel.naam<br>FROM spelers<br>INNER JOIN (<br>SELECT functie AS naam, spelersnr<br>FROM bestuursleden<br>UNION<br>SELECT divisie AS naam, spelersnr<br>FROM teams<br>) AS rommel USING (spelersnr)<br>ORDER BY 1,2;   |  |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer van voor naar achter. Toon 3 getallen na de komma, maximaal 2 voor de komma; gebruik een cast functie.                                  | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe-avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1, 2, 3, 4  |  |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef voor elke mannelijke speler wiens naam minstens 2 keer de letter 'e' bevat zijn functie die hij op dit moment uitoefent, als die er op dit moment één heeft. Sorteer op naam en functie.   | select naam, geslacht, functie<br>from spelers left outer join bestuursleden<br>on spelers.spelersnr = bestuursleden.spelersnr<br>and bestuursleden.eind_datum is null<br>where geslacht = 'M'<br>and naam like '%e%e%'<br>ORDER BY naam, functie;  |  |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer op spelersnr. Zet het berekende verschil om naar het datatype numeric met precisie 5 en schaal 3.                                       | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe-avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1   |  |
| 2023 | Databanken: DB2<br>Examenvraag A | Toon alle mogelijke combinaties van de letters 'x' en 'y'. Tip zie handboek, voorbeeld met cijfers. (Wat is een cartesisch product?). Sorteer   | SELECT A.letter    B.letter<br>FROM (SELECT 'x' AS letter UNION SELECT 'y') as A,<br>(SELECT 'x' AS letter UNION SELECT 'y') as B<br>ORDER BY 1   |  |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef alle spelers die alfabetisch (dus naam en voorletters, in deze volgorde) voor speler 8 staan. Sorteer van voor naar achter. Probeer zo goed of beter te doen dan "Sort (cost=24.31..24.47 rows=67 width=88)"   | SELECT spelersnr, naam, voorletters, geb_datum<br>FROM spelers<br>WHERE naam  voorletters <<br>(SELECT naam  voorletters<br>FROM spelers<br>WHERE spelersnr=8)<br>ORDER BY 1,2,3;   |  |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef alle spelers voor wie meer boetes zijn betaald dan dat ze wedstrijden hebben gespeeld. Zorg dat spelers zonder wedstrijd ook getoond worden. Sorteer van voor naar achter, oplopend.   | SELECT s.naam, s.voorletters, s.geb_datum<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr, s.naam, s.voorletters, s.geb_datum<br>HAVING count(*) ><br>(<br>SELECT count(*)  |  |



|      |                                     |  |   |   |
|------|-------------------------------------|--|---|---|
|      |                                     |  | FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr)<br>ORDER BY 1,2,3  |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef alle spelers die geen enkele wedstrijd voor team 1 hebben gespeeld. Sorteer op naam, daarna op spelersnr.   | SELECT spelersnr, naam<br>FROM spelers<br>WHERE spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden<br>WHERE teamnr = 1)<br>ORDER BY 2,1;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef de spelersgegevens van de speler(s) met het hoogste bedrag (voor één boete, niet het totaalbedrag). Als twee spelers een even hoge boete gehad hebben, moeten beide spelers getoond worden (LIMIT is dus geen optie). Sorteer alfabetisch op naam en voorletters.   | SELECT spelers.spelersnr, voorletters, naam<br>FROM spelers<br>INNER JOIN boetes using (spelersnr)<br>WHERE boetes.bedrag = (<br>SELECT MAX(bedrag)<br>FROM boetes<br>)<br>ORDER BY naam, voorletters   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef een overzicht van de boetebedragen, aantal gewonnen en verloren sets en aantal verschillende functies. Bekijk de output voor de manier hoe het getoond moet worden. Sorteer van links naar rechts. Tip: Het lijkt onlogisch, maar zelfs NULL krijgt een datatype en kan niet impliciet wijzigen van datatype.   | SELECT SUM(bedrag) AS boetebedrag, CAST(null AS int)<br>AS aantalgewonnen, CAST(null AS int) AS aantalverloren,<br>CAST(null AS int) AS aantalfuncties<br>FROM boetes<br>UNION<br>SELECT null AS boetebedrag, SUM(gewonnen) AS<br>aantalgewonnen, null AS aantalverloren, NULL AS<br>aantalfuncties<br>FROM wedstrijden<br>UNION<br>SELECT null AS boetebedrag, null AS aantalgewonnen,<br>SUM(verloren) AS aantalverloren, NULL AS aantalfuncties<br>FROM wedstrijden<br>UNION<br>SELECT null AS boetebedrag, null AS aantalgewonnen,<br>null AS aantalverloren, COUNT(DISTINCT functie) AS<br>aantalfuncties<br>FROM bestuursleden<br>ORDER BY 1,2,3,4; |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef de twee laagste bondnrs terug. (tip: dwz er zijn dus minder dan 2 bondsnr die kleiner zijn) Sorteer van voor naar achter.   | SELECT s.bondsnr<br>FROM spelers s<br>WHERE 2 ><br>(<br>SELECT count(*)<br>FROM spelers sub<br>WHERE sub.bondsnr < s.bondsnr)<br>AND s.bondsnr IS NOT NULL<br>order by 1  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef per team de verloren wedstrijden. Zorg dat teams zonder verloren wedstrijden ook in de output verschijnen. Duid per wedstrijd aan of het om een actief bestuurslid gaat. Sorteer op divisie en wedstrijdnummer.   | SELECT teams.teamnr, teams.divisie,<br>wedstrijden.wedstrijdnr, wedstrijden.spelersnr,<br>CASE<br>WHEN bestuursleden.spelersnr IS NULL THEN '-'<br>ELSE 'actief'<br>END AS bestuurslid<br>FROM teams<br>LEFT OUTER JOIN wedstrijden ON teams.teamnr =<br>wedstrijden.teamnr AND wedstrijden.verloren ><br>wedstrijden.gewonnen<br>LEFT OUTER JOIN bestuursleden ON<br>wedstrijden.spelersnr = bestuursleden.spelersnr AND<br>bestuursleden.eind_datum IS NULL<br>ORDER BY divisie, wedstrijden.wedstrijdnr;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Maak een lijst met de spelers (naam van de speler, voorletter en woonplaats) die ooit gespeeld hebben voor een team dat nu in de tweede divisie speelt en waarvoor geen enkele boete betaald werd voor 1 januari 1981. Geen dubbels, sorteer van voor naar achter.   | SELECT DISTINCT s.naam, s.voorletters, s.plaats<br>FROM spelers s NATURAL INNER JOIN wedstrijden w<br>WHERE w.teamnr IN<br>(SELECT t.teamnr<br>FROM teams t<br>WHERE divisie = 'tweede')<br>AND NOT EXISTS<br>(SELECT 2<br>FROM boetes b<br>WHERE b.datum < '1-1-1981'<br>AND b.spelersnr = s.spelersnr)<br>ORDER BY 1,2,3;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | We willen een statistiek van hoeveel wedstrijden de spelers winnen. Geef een lijst met aantal gewonnen wedstrijden en het aantal spelers dat dit aantal wedstrijden gewonnen heeft. Dus bv als er vier spelers zijn die elk drie wedstrijden hebben gewonnen, dan is de output: aantal_gewonnen: 3, aantal_spelers: 4. Dit graag voor alle aantallen gewonnen wedstrijden en alle spelers. Sorteer op aantal gewonnen wedstrijden. | SELECT aantal_gewonnen, count(spelersnr) AS<br>aantal_spelers<br>FROM (<br>SELECT spelersnr, count(*) AS aantal_gewonnen<br>FROM wedstrijden<br>WHERE gewonnen > verloren<br>GROUP BY spelersnr) AS gewonnen<br>GROUP BY aantal_gewonnen<br>ORDER BY aantal_gewonnen;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef alle spelers die meer wedstrijden gespeeld hebben dan het aantal wedstrijden dat de huidige voorzitter heeft verloren. De huidige voorzitter komt zelf niet in de lijst voor. Gebruik geen subqueries. Sorteer op spelersnr.  | SELECT spelers.spelersnr<br>FROM spelers<br>INNER JOIN wedstrijden ON spelers.spelersnr =<br>wedstrijden.spelersnr,<br>bestuursleden<br>INNER JOIN wedstrijden voorzitterswedstrijden ON<br>bestuursleden.spelersnr = voorzitterswedstrijden.spelersnr<br>AND voorzitterswedstrijden.gewonnen <<br>voorzitterswedstrijden.verloren<br>WHERE bestuursleden.eind_datum IS NULL AND<br>bestuursleden.functie = 'Voorzitter' AND<br>bestuursleden.spelersnr <> spelers.spelersnr<br>GROUP BY spelers.spelersnr<br>HAVING COUNT(DISTINCT wedstrijden.wedstrijdnr) ><br>COUNT(DISTINCT voorzitterswedstrijden.wedstrijdnr)<br>ORDER BY spelers.spelersnr;       |  |
|      |                                     |  | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe-<br>avg as numeric(5,3))  |   |











|      |                                  |   |  |   |
|------|----------------------------------|---|--|---|
| 2023 | Databanken: DB2<br>Examenvraag A | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer op spelersnr. Toon 3 getallen na de komma, zet het verschil om naar het numeric type met precisie van 5 en een schaal van 3.  | FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1  |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef voor elke speler die ooit een boete heeft betaald, de hoogste boete weer en hoelang het geleden is dat deze boete werd betaald. Sorteer van groot naar klein op bedrag en daarna omgekeerd op "leeftijd.." van de boete.   | SELECT sub.spelersnr, sub.bedrag, age(b.datum)<br>FROM (<br>SELECT s.spelersnr, max(b.bedrag) AS bedrag<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr) AS sub<br>INNER JOIN boetes b USING (spelersnr)<br>WHERE sub.bedrag = b.bedrag<br>ORDER BY 2 DESC, 3  |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef per team het hoogste wedstrijdnummer van een wedstrijd, gespeeld door een bestuurslid (actief en niet meer actief) die geen boete heeft gekregen. Sorteer op teamnr.   | SELECT teams.teamnr, MAX(wedstrijden.wedstrijdnr) AS<br>laatstewedstrijd<br>FROM teams<br>INNER JOIN wedstrijden on teams.teamnr =<br>wedstrijden.teamnr<br>INNER JOIN bestuursleden ON wedstrijden.spelersnr =<br>bestuursleden.spelersnr<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr =<br>boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY teams.teamnr<br>ORDER BY teams.teamnr;   |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef een lijst met alle spelersnrs, naam en het aantalwedstrijden ze gespeeld hebben en op een nieuwe lijn het aantal bestuursfuncties die ze hebben/hadden. Spelers die zowel wedstrijden gespeeld hebben als bestuurslid zijn, komen dus twee keer voor in het resultaat. Sorteer op spelersnr en aantal. Geen dubbels tonen.   | SELECT spelers.spelersnr AS nr, spelers.naam,<br>COUNT(*) AS aantal<br>FROM spelers<br>INNER JOIN wedstrijden USING(spelersnr)<br>GROUP BY spelers.spelersnr, spelers.naam<br>UNION<br>SELECT bestuursleden.spelersnr AS nr, spelers.naam,<br>COUNT(*) AS aantal<br>FROM bestuursleden<br>INNER JOIN spelers USING(spelersnr)<br>GROUP BY bestuursleden.spelersnr, spelers.naam<br>ORDER BY nr, aantal;  |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef alle spelers voor wie meer boetes zijn betaald dan dat ze wedstrijden hebben gespeeld. Sorteer van voor naar achter  | SELECT s.naam, s.voorletters, s.geb_datum<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr, s.naam, s.voorletters,<br>s.geb_datum<br>HAVING count(*) ><br>(<br>SELECT count(*)<br>FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr)<br>order by 1,2,3  |    |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef een lijst van alle spelers die bestuurslid geweest zijn (of nu nog zijn) en/of een boete hebben gehad hun aantal boetes en hun laatst begonnen bestuursfunctie. Zorg dat spelers die boetes gehad hebben en bestuurder zijn (geweest) twee keer voorkomen in de kolom 'data' twee verschillende waardes. Sorteer op naam, voorletters en data. Gebruik de to_char functie voor het formaat van de geboortedatum (bv 12/12/1900). | SELECT naam, voorletters, to_char(geb_datum,<br>'DD/MM/YYYY') AS geboortedatum, data.data<br>FROM spelers<br>INNER JOIN (<br>SELECT spelersnr, 'Boetes: '    CAST(COUNT(*) AS<br>CHARACTER(20)) AS data<br>FROM boetes<br>GROUP BY spelersnr<br>UNION<br>SELECT spelersnr, functie AS data<br>FROM bestuursleden<br>WHERE begin_datum = (<br>SELECT MAX(begin_datum)<br>FROM bestuursleden alle<br>WHERE alle.spelersnr = bestuursleden.spelersnr<br>)<br>) AS data ON spelers.spelersnr = data.spelersnr<br>ORDER BY naam, voorletters, data; |  |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef voor elke mannelijke speler wiens naam minstens 2 keer de letter 'e' bevat een lijst van de functies die hij op dit moment uitoefent. Ook mannelijke spelers zonder huidige functie moeten getoond worden. Sorteer op spelersnr.   | select naam, geslacht, functie<br>from spelers left outer join bestuursleden<br>on spelers.spelersnr = bestuursleden.spelersnr<br>and bestuursleden.eind_datum is null<br>where geslacht = 'M'<br>and naam like '%e%e%'<br>ORDER BY spelers.spelersnr;   |  |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef een lijst van alle huidige bestuursleden die nog geen boete gekregen hebben, maar wel al minstens één wedstrijd verloren hebben. Sorteer op spelersnr en wedstrijdnr.  | SELECT S.spelersnr, S.naam, B.functie, B.begin_datum,<br>W.wedstrijdnr<br>FROM spelers S<br>INNER JOIN bestuursleden B ON S.spelersnr =<br>B.spelersnr AND B.eind_datum IS NULL<br>INNER JOIN wedstrijden W ON S.spelersnr = W.spelersnr<br>AND W.gewonnen < W.verloren<br>LEFT OUTER JOIN boetes BO ON S.spelersnr =<br>BO.spelersnr<br>WHERE BO.betalingsnr IS NULL<br>ORDER BY S.spelersnr, W.wedstrijdnr   |  |
| 2023 | Databanken: DB2<br>Examenvraag A | Geef het gemiddeld boetebedrag per speler, afgerond op twee cijfers na de komma. spelers zonder boete moeten bovenaan staan met 'geen boetes' Sorteer daarna op spelersnaam en gemiddeld boetebedrag. Om een waarde om te zetten naar een ander datatype, kan je ofwel CAST() gebruiken ofwel de TO_CHAR() met 'FM999.00' als opmaakmasker.   | SELECT spelers.naam,<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 'geen boetes'<br>ELSE CAST(ROUND(AVG(boetes.bedrag), 2) AS<br>varchar(8))<br>END<br>AS gemiddeld<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr =<br>boetes.spelersnr<br>GROUP BY spelers.spelersnr, spelers.naam<br>ORDER BY<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 0<br>ELSE 1<br>END, spelers.naam, AVG(boetes.bedrag);   |  |

|      |                                     |  |  |  |
|------|-------------------------------------|--|--|--|
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef per team de verloren wedstrijden. Zorg dat teams zonder verloren wedstrijden ook in de output verschijnen.<br>Duid per wedstrijd aan of het om een actief bestuurslid gaat.<br>Sorteer op divisie en wedstrijdnummer.   | SELECT teams.teamnr, teams.divisie,<br>wedstrijden.wedstrijdnr, wedstrijden.spelersnr,<br>CASE<br>WHEN bestuursleden.spelersnr IS NULL THEN '-'<br>ELSE 'actief'<br>END AS bestuurslid<br>FROM teams<br>LEFT OUTER JOIN wedstrijden ON teams.teamnr =<br>wedstrijden.teamnr AND wedstrijden.verloren ><br>wedstrijden.gewonnen<br>LEFT OUTER JOIN bestuursleden ON<br>wedstrijden.spelersnr = bestuursleden.spelersnr AND<br>bestuursleden.eind_datum IS NULL<br>ORDER BY divisie, wedstrijden.wedstrijdnr;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef een lijst van alle spelers (spelersnr en woonplaats) die met minstens twee in dezelfde plaats wonen. Sorteer aflopend op woonplaats, daarna op spelersnr.   | SELECT spelersnr, plaats<br>FROM spelers<br>WHERE plaats IN (<br>SELECT plaats<br>FROM spelers<br>GROUP BY plaats<br>HAVING COUNT(*) >= 2<br>)<br>ORDER BY plaats DESC, spelersnr;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef de naam en het spelersnummer van de spelers die ooit penningmeester geweest zijn van de club, die bovendien ooit een boete betaald hebben van meer dan 75 euro, en die ooit een wedstrijd gewonnen hebben met meer dan 2 sets verschil. Sorteer van voor naar achter.<br>Probeer gelijk of beter te doen dan "Unique (cost=100.38..100.54 rows=21 width=68)".   | SELECT DISTINCT s.naam, s.spelersnr<br>FROM ((spelers s INNER JOIN bestuursleden be USING (spelersnr)) INNER JOIN boetes bo USING (spelersnr))<br>INNER JOIN wedstrijden w USING (spelersnr)<br>WHERE bo.bedrag >= 75<br>AND be.functie = 'Penningmeester'<br>AND be.eind_datum IS NOT NULL<br>AND (w.gewonnen - w.verloren) > 1<br>ORDER BY 1,2   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding. Sorteer van voor naar achter.<br>Probeer gelijk of beter te doen dan "Sort (cost=33.16..33.66 rows=200 width=86)".   | SELECT s.spelersnr,s.naam,s.voorletters,s.jaartoe-<br>(SELECT avg(jaartoe)<br>FROM spelers) AS Verschil<br>FROM spelers s<br>ORDER BY 1,2,3,4  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | We zijn wat betreft boetes alleen geïnteresseerd als de speler geboren is voor 1963.<br><br>Verder uitleg:<br>Geef een lijst van ALLE spelers die bestuurslid geweest zijn en/of een boete hebben gehad, (alleen als ze geboren zijn voor 1963) en hun laatst begonnen bestuursfunctie.<br>Zorg ervoor dat spelers die een bestuursfunctie hebben/hadden, geboren zijn voor 1963, maar geen boete hebben gekregen, toch in het resultaat voorkomen met een extra lijn 'Geen boetes'.<br>Spelers die geen bestuurslid zijn geweest, maar een boete hebben gehad, komen één keer voor in het resultaat met hun aantal boetes.<br>Spelers die geen bestuurslid zijn geweest en geboren zijn na 1962, moeten ook één keer in het resultaat voorkomen met als data 'Gewone speler' (het kan zelfs zijn dat geen boete gekregen hebben, alle spelers moeten sowieso getoond worden).<br>Sorteer op naam, voorletters en data (let op de hoofdletters in het veld data).<br><br>Gebruik de to_char functie voor het formaat van de geboortedatum (bv 12/12/1900). | SELECT naam, voorletters, to_char(geb_datum,<br>'DD/MM/YYYY') AS geboortedatum,<br>CASE<br>WHEN data.data IS NULL THEN 'Gewone speler'<br>ELSE data.data<br>END AS data<br>FROM spelers<br>LEFT OUTER JOIN (<br>SELECT spelers.spelersnr,<br>CASE<br>WHEN COUNT(betalingsnr) = 0 THEN 'Geen boetes'<br>ELSE 'Boetes: '    CAST(COUNT(betalingsnr) AS<br>CHARACTER(20))<br>END AS data<br>FROM spelers<br>LEFT OUTER JOIN boetes USING (spelersnr)<br>WHERE EXTRACT(YEAR FROM geb_datum) < 1963<br>GROUP BY spelersnr<br>UNION<br>SELECT spelersnr, functie AS data<br>FROM bestuursleden<br>WHERE begin_datum = (<br>SELECT MAX(begin_datum)<br>FROM bestuursleden alle<br>WHERE alle.spelersnr = bestuursleden.spelersnr<br>)<br>) AS data ON spelers.spelersnr = data.spelersnr<br>ORDER BY naam, voorletters, data; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef van elke boete het betalingsnr, het boetebedrag en het percentage dat het bedrag uitmaakt van de som van alle bedragen. Sorteer deze data op het betalingsnr. Zorg dat er maar twee getallen na de komma getoond worden (rond af). Sorteer van voor naar achter.  | SELECT b1.betalingsnr, b1.bedrag,<br>round (100*b1.bedrag/sum(b2.bedrag),2)<br>FROM boetes b1, boetes b2 GROUP BY b1.betalingsnr,<br>b1.bedrag ORDER BY 1,2,3;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef alle bestuursleden die geboren zijn voor 1970-08-05 en hun staat van dienst.<br>Bekijk de output om de juiste formatering te zien.<br>gebruik een combinatie van CASE, de    functie en datum opmaak.<br>Sorteer op naam en begin datum   | SELECT spelers.voorletters, spelers.naam,<br>bestuursleden.functie, 'actief: '   <br>CASE<br>WHEN bestuursleden.eind_datum IS NULL THEN 'sinds '<br>   to_char(bestuursleden.begin_datum, 'DD/MM/YYYY')<br>ELSE 'van '    to_char(bestuursleden.begin_datum,<br>'DD/MM')    ' tot '    to_char(bestuursleden.eind_datum,<br>'DD/MM/YYYY')<br>END AS periode<br>FROM bestuursleden, spelers<br>WHERE bestuursleden.spelersnr = spelers.spelersnr<br>AND spelers.geb_datum < '1970-08-05'<br>ORDER BY spelers.naam, bestuursleden.begin_datum  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef een overzicht van alle spelers met een bondsnummer.<br>Sorteer als volgt:<br>- Eerst moeten alle spelers uit zoetermeer getoond worden, deze sorteer je op naam. (spelers met dezelfde naam uit zoetermeer worden gesorteerd op bondsnr)<br>- Daarna de andere spelers, gesorteerd op bondsnr.<br><br>Tip: je kan CASE ook gebruiken in de ORDER BY component (zet dan '0')   | SELECT spelersnr, naam, plaats, bondsnr<br>FROM spelers<br>WHERE bondsnr IS NOT NULL<br>ORDER BY<br>CASE<br>WHEN plaats = 'Zoetermeer' THEN '0'<br>ELSE bondsnr<br>END, naam, bondsnr;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef van elke speler die enkel wedstrijden gewonnen heeft voor team nr 1 en voor wie in totaal meer dan 100 euro aan boete betaald is, het spelersnummer, zijn naam, woonplaats en het totale boetebedrag.<br>Dit resultaat moet aflopend geordend worden op het totale boetebedrag. Sorteer van voor naar achter.   | SELECT s.spelersnr, s.naam, s.plaats, sum(b.bedrag)<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>WHERE s.spelersnr IN (<br>SELECT spelersnr<br>FROM wedstrijden w<br>WHERE gewonnen > verloren<br>AND teamnr=1)<br>GROUP BY s.spelersnr, s.naam, s.plaats  |  |

|      |                                     |   |  |   |
|------|-------------------------------------|---|--|---|
|      |                                     |   | HAVING sum(b.bedrag)>100<br>order by 1,2,3,4   |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef van elke speler die minstens 1 wedstrijd gewonnen heeft voor team nr 1 en voor wie in totaal meer dan 100 euro aan boete betaald is, het spelersnummer, zijn naam, woonplaats en het totale boetebedrag. Sorteer van voor naar achter.   | SELECT s.spelersnr, s.naam, s.plaats, sum(b.bedrag)<br>FROM spelers s INNER JOIN boetes b on<br>(b.spelersnr=s.spelersnr)<br>WHERE s.spelersnr IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden w<br>WHERE gewonnen > verloren<br>AND teamnr=1)<br>GROUP BY s.spelersnr, s.naam, s.plaats<br>HAVING sum(b.bedrag)>100<br>order by 1,2,3,4;  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Selecteer de namen van voorzitters en de periode (= begindatum streepje einddatum) waarin ze voorzitter waren (of nog steeds zijn) en op welke datum ze beboet zijn geweest. Gebruik   .  | select s.naam, CASE WHEN be.eind_datum IS NOT<br>NULL THEN to_char(be.begin_datum, 'DD/MM/YYYY')    ' - '    to_char(be.eind_datum, 'DD/MM/YYYY') ELSE<br>to_char(be.begin_datum, 'DD/MM/YYYY')    ' - ' END as<br>periode, bo.datum<br>from bestuursleden as be, boetes as bo, spelers as s<br>where be.spelersnr = bo.spelersnr<br>and functie = 'Voorzitter'<br>and s.spelersnr = be.spelersnr                          |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Sorteer de teams in functie van het aantal verloren wedstrijden (oplopend). Als het aantal verloren wedstrijden gelijk is, sorteer je op het totaal aantal gewonnen sets (aflopend) en op divisienaam (oplopend). Een wedstrijd die verloren werd met 3 - 0 telt niet mee in deze output. Geef het aantal wedstrijden dat voldoet aan bovenstaande voorwaarde mee in de output.   | SELECT teams.divisie, COUNT(*) AS aantal<br>FROM wedstrijden, teams<br>WHERE wedstrijden.teamnr = teams.teamnr<br>AND wedstrijden.verloren > wedstrijden.gewonnen<br>AND NOT (wedstrijden.verloren = 3 AND<br>wedstrijden.gewonnen = 0)<br>GROUP BY teams.divisie<br>ORDER BY aantal, SUM(wedstrijden.gewonnen) DESC,<br>teams.divisie;  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef voor alle huidige bestuursleden hun functie en de lijst van boetes die voor hen werd betaald. Omdat je dit wil vergelijken met de boetebedragen die betaald werden voor spelers die niet in het bestuur zitten, wil je deze boetebedragen ook opnemen in de tweede kolom van je resultaat. Sorteer je antwoord eerst op functie en daarna op het boetebedrag.  | SELECT functie, bedrag<br>FROM boetes BT FULL OUTER JOIN bestuursleden B<br>ON (BT.spelersnr = B.spelersnr<br>AND eind_datum is null)<br>ORDER BY 1,2;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef een overzicht van alle spelers, gevolgd door alle bestuursleden, gesorteerd op jaar van toetreding of beginjaar van hun functie en vervolgens op spelersnr. Geen dubbels tonen.  | SELECT spelersnr AS veld1, naam AS veld2, jaartoe AS<br>veld3<br>FROM spelers<br>UNION<br>SELECT spelersnr AS veld1, functie AS veld2, EXTRACT<br>(YEAR FROM begin_datum) AS veld3<br>FROM bestuursleden<br>ORDER BY veld3, veld1  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef een lijst van alle mogelijke dubbelcombinaties van spelers. Beperking: de verschillende spelers mogen geen kapitein zijn van hetzelfde team (moest dit mogelijk zijn). Orden eerst naar speler1, dan naar speler2.   | select DISTINCT s.voorletters    ' '    s.naam AS "speler1",<br>t.voorletters    ' '    t.naam AS "speler2"<br>from spelers as s, spelers as t, teams as te, teams as tt<br>where s.spelersnr <> t.spelersnr AND<br>NOT (s.spelersnr = te.spelersnr AND t.spelersnr =<br>tt.spelersnr AND te.teamnr = tt.teamnr)<br>ORDER BY 1,2   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef het gemiddeld boetebedrag per speler, afgerond op twee cijfers na de komma. spelers zonder boete moeten bovenaan staan met 'geen boetes' Sorteer daarna op spelersnaam en vervolgens op gemiddeld boetebedrag. Gebruik als datatype van de 2de kolom varchar(8) voor spelers die wel boetes hebben.  | SELECT spelers.naam,<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 'geen boetes'<br>ELSE CAST(ROUND(AVG(boetes.bedrag), 2) AS<br>varchar(8))<br>END<br>AS gemiddeld<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr =<br>boetes.spelersnr<br>GROUP BY spelers.spelersnr, spelers.naam<br>ORDER BY<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 0<br>ELSE 1<br>END, spelers.naam, AVG(boetes.bedrag);   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag A | Geef per team de leeftijd van de aanvoerder (tip: postgresql heeft een AGE() functie) en het aantal verschillende spelers dat voor dit team gespeeld heeft. Alleen teams waarvoor wedstrijden zijn gespeeld en die een aanvoerder hebben, moeten vermeld worden. Sorteer op leeftijd en daarna op aantal verschillende spelers en daarna op teamnr.   | SELECT teams.teamnr, CAST(EXTRACT(year from<br>AGE(NOW(), spelers.geb_datum)) AS varchar(2))    ' jaar'<br>AS leeftijd, COUNT(DISTINCT wedstrijden.spelersnr) AS<br>aantalspelers<br>FROM teams<br>INNER JOIN spelers ON teams.spelersnr =<br>spelers.spelersnr<br>INNER JOIN wedstrijden ON teams.teamnr =<br>wedstrijden.teamnr<br>GROUP BY teams.teamnr, spelers.geb_datum<br>ORDER BY leeftijd, aantalspelers, teamnr; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef voor elke een klant een overzicht aan uitgaven. Hoe? Geef voor elke klant een cumulatief overzicht van prijs van de reizen waar hij aan deelgenomen heeft. De volgorde, waarin er wordt cumulatief wordt opgeteld, wordt bepaald door de vertrekdatum van de reis. Sorteer van voor naar achter. Tip: vergelijk deze uitvoer met de uitvoer van een query die een gesorteerd overzicht geeft van de klanten en hun deelnames aan reizen. | select klantnr, reisnr, sum(prijs) OVER (partition by klantnr<br>order by vertrekdatum)<br>from reizen natural join deelnames<br>order by 1, 2, 3;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef het aantal unieke satellieten van planeten binnen ons sterrenstelsel   | SELECT count(DISTINCT S.objectnaam) as<br>aantal_satellieten<br>FROM hemelobjecten Z, hemelobjecten P, hemelobjecten<br>S<br>WHERE Z.satellietvan is null AND P.satellietvan =<br>Z.objectnaam AND S.satellietvan = P.objectnaam   |  |
|      |                                     |   | SELECT MAX(manen.diameter) AS grootstemaan<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten manen ON   |   |

|      |                                     |   |  |  |
|------|-------------------------------------|---|--|--|
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de diameter van de grootste, niet bezochte maan (satelliet van een planeet).   | planeten.objectnaam = manen.satellietvan AND<br>planeten.satellietvan = 'Zon'<br>LEFT OUTER JOIN bezoeken ON manen.objectnaam = bezoeken.objectnaam<br>WHERE bezoeken.reisnr IS NULL;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Vind per planeet die zijn maan die het verst ervan verwijderd is. Geef planeet, maan en afstand en sorteer stijgend op afstand.   | SELECT planeet.objectnaam as planeet, M.objectnaam as maan, M.afstand as afstand<br>FROM hemelobjecten ster<br>INNER JOIN hemelobjecten planeet ON ster.satellietvan IS NULL AND planeet.satellietvan = ster.objectnaam<br>INNER JOIN hemelobjecten maan ON maan.satellietvan = planeet.objectnaam<br>INNER JOIN hemelobjecten M on M.satellietvan = planeet.objectnaam<br>GROUP BY planeet.objectnaam, M.objectnaam<br>HAVING M.afstand = MAX(maan.afstand)<br>ORDER BY afstand   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Hoeveel kilometers heeft iedereen in totaal gevlogen tot nu toe en hoeveel hebben ze hier in totaal voor betaald. Vermits we de posities van de planeten niet kennen, mag je de afstanden van de hemelobjecten direct gebruiken. Geef het totaal gependeerde bedrag, de afgelegde kilometers, de prijs per kilometer en datum van hun laatste vlucht van al hun persoonlijke reizen. In het geval dat iemand niet op reis is geweest of geen kilometers gedaan heeft, toon je de boodschap 'veel geld voor niks of niet op reis geweest' in de kolom prijs_per_kilometer. Sorteer van voor naar achter. | SELECT k.klantnr, k. naam    ' '    k.vnaam AS naam, tot _bedrag, tot _afstand, CASE WHEN tot _afstand <> 0 THEN CAST(tot _bedrag/tot _afstand AS varchar) ELSE 'veel geld voor niks of niet op reis geweest' END AS prijs_per _kilometer, laatste _reis _datum FROM klanten k NATURAL LEFT OUTER JOIN ((SELECT klantnr, sum(prijs) AS tot _bedrag, max(vertrekdatum) AS laatste _reis _datum FROM deelnames NATURAL LEFT OUTER JOIN reizen GROUP BY klantnr) AS dr NATURAL LEFT OUTER JOIN (SELECT klantnr, sum(afstand) AS tot _afstand FROM (deelnames NATURAL INNER JOIN bezoeken) NATURAL INNER JOIN hemelobjecten GROUP BY klantnr) AS bh) ORDER BY 1,2,3,4,5,6; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de deelnemers waarbij hun aantal reizen die ze ondernemen groter is dan alle hemelobjecten (die niet beginnen met de letter 'M') hun aantal keren dat ze bezocht zijn. Of anders geformuleerd: Geef de deelnemers met meer deelnames dan het grootste aantal bezoeken aan een hemelobject dat niet met de letter 'M' begint (:deze deelnemer meer deelnames heeft dan de "grootste" .. = deze deelnemer heeft meer deelnames dan "alle" ..) Sorteer op klantnr.  | SELECT klantnr, vnaam, naam, COUNT(*) AS aantaldeelnames FROM klanten INNER JOIN deelnames using (klantnr) GROUP BY klantnr, vnaam, naam HAVING COUNT(*) > ALL ( SELECT COUNT(*) FROM bezoeken WHERE objectnaam NOT LIKE 'M%' GROUP BY objectnaam ) ORDER BY klantnr,2,3,4;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef voor elke geboortejaar van de klanten, het aantal klanten, het kleinste klantennummer en het grootste klantennummer. Geef ook het totaal aantal klanten en het kleinste en grootste klantennummer. Sorteer van voor naar achter.   | SELECT extract(year from geboortedatum), count(*),min(klantnr),max(klantnr) FROM klanten GROUP BY ROLLUP (extract(year from geboortedatum)) ORDER BY 1,2,3,4;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef het reisnr, de prijs en vertrekdatum van de reis met de hoogste gemiddelde verblijfsduur op een hemelobject (=som van de verblijfsduur / aantal bezoeken per reis).  | SELECT reizen.reisnr, prijs, vertrekdatum FROM reizen INNER JOIN bezoeken USING (reisnr) GROUP BY reizen.reisnr, prijs, vertrekdatum HAVING (SUM(verblijfsduur) / COUNT(objectnaam)) = ( SELECT MAX(gemiddelde) FROM ( SELECT SUM(verblijfsduur) / COUNT(objectnaam) as gemiddelde FROM bezoeken GROUP BY reisnr ) AS reisgemiddelden )  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de naam, diameter en het langste verblijf op van alle hemelobjecten met minder dan 5 hemelobjecten die rond dit object draaien. (ze moeten dus bezocht zijn) Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam, hemelobjecten.diameter, MAX(bezoeken.verblijfsduur) AS maximale _verblijf FROM bezoeken INNER JOIN hemelobjecten ON bezoeken.objectnaam = hemelobjecten.objectnaam LEFT OUTER JOIN hemelobjecten manen ON hemelobjecten.objectnaam = manen.satellietvan GROUP BY hemelobjecten.objectnaam, hemelobjecten.diameter HAVING COUNT(DISTINCT manen.objectnaam) < 5 ORDER BY objectnaam;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Maak een lijst van de mensen die Mars wel bezocht hebben maar lo nog niet. Sorteer van voor naar achter.  | SELECT k.klantnr, k. naam FROM ((klanten k INNER JOIN deelnames d using (klantnr)) INNER JOIN reizen r using(reisnr)) INNER JOIN bezoeken b using(reisnr) WHERE b.objectnaam = 'Mars' EXCEPT SELECT k.klantnr, k. naam FROM ((klanten k INNER JOIN deelnames d using (klantnr)) INNER JOIN reizen r using(reisnr)) INNER JOIN bezoeken b using(reisnr) WHERE b.objectnaam = 'Io' order by 1,2  |  |
|      |                                     |   | SELECT planeten.objectnaam FROM hemelobjecten planeten LEFT OUTER JOIN hemelobjecten satellieten ON planeten.objectnaam = satellieten.satellietvan WHERE planeten.satellietvan = 'Zon' GROUP BY planeten.objectnaam HAVING COUNT(*) = (  |  |



|      |                                     |  |  |   |
|------|-------------------------------------|--|--|---|
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de planeet (draait dus rond de zon) met de meeste satellieten.<br>Sorteer op objectnaam.  | <pre>SELECT MAX(aantalsatellieten) FROM ( SELECT COUNT(*) AS aantalsatellieten FROM hemelobjecten planeten LEFT OUTER JOIN hemelobjecten satellieten ON planeten.objectnaam = satellieten.satellietvan WHERE planeten.satellietvan = 'Zon' GROUP BY planeten.objectnaam ) AS aantallen ORDER BY planeten.objectnaam</pre>  |     |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef voor elk hemelobject de minimale en maximale gemiddelde afstand tot zijn zon (de centrale ster in een sterrenstelsel) als je weet dat de kolom 'afstand' de gemiddelde afstand bevat tot het hemelobject waar rond ze draaien. Met de grootte van het hemelobject hoeft je geen rekening te houden.<br>Sorteer op minimale afstand en op objectnaam.<br>Tip: bekijk de functie COALESCE om met null-waardes rekening te houden. | <pre>SELECT hemelobjecten.objectnaam, coalesce(hemelobjecten.afstand, 0) + coalesce(draaitrond.afstand, 0) AS maximale_afstand, abs(coalesce(draaitrond.afstand, 0) - coalesce(hemelobjecten.afstand, 0)) AS minimale_afstand FROM hemelobjecten LEFT OUTER JOIN hemelobjecten draaitrond ON draaitrond.objectnaam = hemelobjecten.satellietvan ORDER BY minimale_afstand, objectnaam;</pre> |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de diameter van het grootste hemellichaam dat bezocht is op de vroegste reis waar klantnr 126 niet op meegegaan is.   | <pre>SELECT MAX(diameter) AS grootste FROM hemelobjecten INNER JOIN bezoeken USING (objectnaam) INNER JOIN reizen USING (reisnr) WHERE vertrekdatum = ( SELECT MIN(vertrekdatum) FROM reizen INNER JOIN bezoeken USING (reisnr) WHERE reisnr NOT IN ( SELECT reisnr FROM deelnames WHERE klantnr = 126 ) )</pre>   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de hemelobjecten die nog nooit bezocht zijn (een bezoek heeft een verblijfsduur van minstens één dag).  | <pre>SELECT hemelobjecten.objectnaam FROM hemelobjecten LEFT OUTER JOIN bezoeken ON hemelobjecten.objectnaam = bezoeken.objectnaam AND bezoeken.verblijfsduur &gt; 0 WHERE bezoeken.objectnaam IS NULL ORDER BY hemelobjecten.objectnaam;</pre>  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | We willen telkens het aantal reizen en de totale prijs van de volgende situaties. Voor elke maand van vertrek, voor elk tiental van de reisduur, voor de combinatie van maand van vertrek en tiental van de reisduur en het totale plaatje.<br>Sorteer van voor naar achter.   | <pre>SELECT extract(month from vertrekdatum),round(reisduur/10), count(*), sum(prijs) FROM reizen GROUP BY CUBE (extract(month from vertrekdatum),round(reisduur/10)) ORDER by 1,2,3,4;</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Sorteer de klanten aflopend op gemiddelde kostprijs per bezoek (totaalprijs van alle reizen per klant/totaal aantal dagen verblijf op een hemelobject), op twee cijfers na de komma afgerond, daarna op klantnr.   | <pre>SELECT klanten.klantnr, ROUND(SUM(reizen.prijs)/SUM(bezoeken.verblijfsduur),2) AS gemiddeldeprijs FROM klanten INNER JOIN deelnames ON klanten.klantnr = deelnames.klantnr INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr INNER JOIN bezoeken ON reizen.reisnr = bezoeken.reisnr GROUP BY klanten.klantnr ORDER BY gemiddeldeprijs DESC, klantnr;</pre>                          |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef het hemellichaam dat het laatst bezocht is.<br>Gebruik hiervoor de laatste vertrekdatum van de reis en laatste volgnummer van bezoek. Tip: gebruik hiervoor een rij-subquery.<br>Gebruik geen limit of top.   | <pre>SELECT objectnaam FROM bezoeken WHERE (reisnr, volgnr) = ( SELECT bezoeken.reisnr, MAX(bezoeken.volgnr) FROM bezoeken INNER JOIN reizen laatstereis USING (reisnr) WHERE laatstereis.vertrekdatum = ( SELECT MAX(vertrekdatum) FROM reizen ) ) GROUP BY bezoeken.reisnr )</pre>   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef voor de hemelobjecten een diameter groter dan 1000 het aantal hemelobjecten en de gemiddelde diameter per groep die aan het volgende voldoet. We zien per object zien hoeveel satellieten hieraan voldoen, daarnaast in combinatie met hetzelfde object en afstand per 100tal. Alsook een algemeen overzicht.<br>Sorteer van voor naar achter.  | <pre>SELECT objectnaam, round(afstand/100),satellietvan, count(*), avg(diameter) FROM hemelobjecten WHERE diameter &gt; 1000 GROUP BY ROLLUP (satellietvan, (objectnaam,round(afstand/100)));</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Hoe lang was het geleden dat er nog een reis vertrokken was?<br>Geef daarnaast de totale reisduur per jaar incrementeel in de tijd (hier genaamd jaar_duur).<br>Sorteer op reisnr en de andere kolommen.   | <pre>SELECT reisnr, lag(reisnr) OVER w1 as vorig_reisnr, vertrekdatum, vertrekdatum - lag(vertrekdatum) OVER w1 as tussen_tijd, reisduur, extract(year from vertrekdatum) as jaar, sum(reisduur) OVER (partition by extract(year from vertrekdatum) order by vertrekdatum) as jaar_duur FROM reizen WINDOW w1 as (order by vertrekdatum) ORDER BY 1,2,3,4,5,6,7;</pre>                       |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Maak een lijst van klanten die meer dan 2 keer een reis gemaakt hebben waarbij er geen bezoek was aan Jupiter.   | <pre>SELECT k.klantnr, (k. naam    ' '    k.vnaam) AS klantnaam, COUNT(d.reisnr) AS aantalreizen FROM klanten k INNER JOIN deelnames d USING (klantnr) WHERE d.reisnr NOT IN (SELECT b.reisnr FROM bezoeken b INNER JOIN hemelobjecten o ON b.objectnaam = o.objectnaam WHERE o.objectnaam = 'Jupiter') GROUP BY k.klantnr, k. naam, k.vnaam</pre>   |  |









|      |                                  |   |  |   |
|------|----------------------------------|---|--|---|
|      |                                  |   | HAVING count(d.reisnr) > 2;  |   |
| 2023 | Databanken: DB2<br>Examenvraag B | Geef een lijst van alle hemelobjecten die meer keer bezocht gaan worden dan Jupiter. (onafhankelijk van het aantal deelnames)<br>Sorteer op objectnaam.   | SELECT hemelobjecten.objectnaam, hemelobjecten.diameter<br>FROM hemelobjecten<br>INNER JOIN bezoeken USING (objectnaam)<br>GROUP BY hemelobjecten.objectnaam, hemelobjecten.diameter<br>HAVING COUNT(*) > (<br>SELECT COUNT(*)<br>FROM bezoeken<br>WHERE objectnaam = 'Jupiter'<br>)<br>ORDER BY hemelobjecten.objectnaam  |    |
| 2023 | Databanken: DB2<br>Examenvraag B | Geef de klantnr voor de klant met het meeste bezoeken aan de maan. Geef ook het aantal bezoeken.<br>Gebruik geen limit of top.  | SELECT d.klantnr, count(b.objectnaam)<br>FROM deelnames d<br>INNER JOIN reizen r ON d.reisnr = r.reisnr<br>INNER JOIN bezoeken b ON r.reisnr = b.reisnr<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr<br>HAVING COUNT(b.objectnaam) = (<br>SELECT MAX(c)<br>FROM (<br>SELECT d.klantnr, COUNT(b.objectnaam) AS c<br>FROM deelnames d<br>INNER JOIN bezoeken b ON d.reisnr = b.reisnr<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr<br>) AS temp<br>);  |    |
| 2023 | Databanken: DB2<br>Examenvraag B | Geef een lijst van alle planeten die minstens 1 satelliet hebben met een diameter groter dan 2000km. Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer op planeetnaam.<br>(to_char 999 om de telling om te zetten)  | SELECT planeet.objectnaam, CASE WHEN count(sat.objectnaam) = 0 THEN 'Geen' ELSE to_char(count(sat.objectnaam), '999') END AS "interessante satellieten"<br>FROM hemelobjecten planeet INNER JOIN hemelobjecten ster ON (planeet.satellietvan = ster.objectnaam AND ster.satellietvan is null)<br>LEFT OUTER JOIN hemelobjecten sat ON (sat.satellietvan = planeet.objectnaam AND sat.diameter > 2000)<br>GROUP BY planeet.objectnaam<br>ORDER BY planeet.objectnaam  |    |
| 2023 | Databanken: DB2<br>Examenvraag B | Geef voor elke reis het aantal klanten waarvan de naam niet met een 'G' begint en waarvan de periode van de geboortedatum van de klant tot de vertrekdatum van de reis overlapt met de huidige datum en 50 jaar verder (gebruik hiervoor de gepaste operator: OVERLAPS).<br>Indien er op de reis hemelobjecten worden bezocht waarvan de tweede letter van het hemelobject voorkomt in de naam van het hemelobject waarvan dit bezocht hemelobject een satelliet is, dan wordt deze reis genegeerd.<br>Sorteer op reisnr. | SELECT r.reisnr, count(k.klantnr)<br>FROM (REIZEN r NATURAL LEFT OUTER JOIN deelnames)<br>LEFT OUTER JOIN klanten k ON (k.klantnr = deelnames.klantnr)<br>AND upper(naam) NOT LIKE 'G%'<br>AND (geboortedatum,vertrekdatum) OVERLAPS (CURRENT_DATE,CURRENT_DATE+interval '50 year')<br>WHERE r.reisnr NOT IN (SELECT reisnr<br>FROM bezoeken NATURAL INNER JOIN hemelobjecten<br>WHERE lower(satellietvan) LIKE '%'    lower(substring(objectnaam FROM 2 FOR 1))    '%')<br>GROUP BY r.reisnr<br>ORDER BY 1,2; |  |
| 2023 | Databanken: DB2<br>Examenvraag B | Geef een lijst van alle planeten die minstens 1 satelliet hebben op een afstand groter dan 500km. Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer omgekeerd alfabetisch op planeetnaam.<br><br>Gebruik als datatype voor de 2de kolom voor het tellen:to_char('999')  | SELECT planeet.objectnaam, CASE WHEN count(sat.objectnaam) = 0 THEN 'Geen' ELSE to_char(count(sat.objectnaam), '999') END AS "ver verwijderde satellieten"<br>FROM hemelobjecten planeet INNER JOIN hemelobjecten ster ON (planeet.satellietvan = ster.objectnaam AND ster.satellietvan is null)<br>LEFT OUTER JOIN hemelobjecten sat ON (sat.satellietvan = planeet.objectnaam AND sat.afstand > 500)<br>GROUP BY planeet.objectnaam<br>ORDER BY planeet.objectnaam desc                                      |  |
| 2023 | Databanken: DB2<br>Examenvraag B | Geef per reis incrementeel de totale verblijfsduur volgens de volgorde waarin de hemelobjecten bezocht worden. Geef ook de totale verblijfsduur van alle reizen om alles in perspectief te zetten.<br>Sorteer op reisnr, volgnr, objectnaam, verblijfsduur, de volgende kolommen.   | SELECT reisnr, volgnr, objectnaam, verblijfsduur, sum(verblijfsduur)<br>OVER(partition by reisnr order by volgnr) as inc_duur, sum(verblijfsduur) over () as tot_duur<br>FROM bezoeken<br>ORDER by 1,2,3,4,5,6;  |  |
| 2023 | Databanken: DB2<br>Examenvraag B | Geef een lijst van alle planeten die minstens 1 satelliet hebben wiens naam een kleine letter i bevat. Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer op planeetnaam.<br>(Gebruikt to_char .. '999') voor omzetting telling)   | SELECT planeet.objectnaam, CASE WHEN count(sat.objectnaam) = 0 THEN 'Geen' ELSE to_char(count(sat.objectnaam), '999') END AS "satellieten met een letter i"<br>FROM hemelobjecten planeet INNER JOIN hemelobjecten ster ON (planeet.satellietvan = ster.objectnaam AND ster.satellietvan is null)<br>LEFT OUTER JOIN hemelobjecten sat ON (sat.satellietvan = planeet.objectnaam AND sat.objectnaam LIKE "%i%")<br>GROUP BY planeet.objectnaam<br>ORDER BY planeet.objectnaam                                  |  |
| 2023 | Databanken: DB2<br>Examenvraag B | Geef de naam, diameter en het langste verblijf op alle hemelobjecten met minder dan 5 hemelobjecten die rond dit object draaien. Geef enkel hemelobjecten die bezocht zijn of waar gepasseerd wordt.<br>Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam, hemelobjecten.diameter, MAX(bezoeken.verblijfsduur) AS maximale_verblijf<br>FROM bezoeken<br>INNER JOIN hemelobjecten ON bezoeken.objectnaam = hemelobjecten.objectnaam<br>LEFT OUTER JOIN hemelobjecten manen ON hemelobjecten.objectnaam = manen.satellietvan<br>GROUP BY hemelobjecten.objectnaam, hemelobjecten.diameter<br>HAVING COUNT(DISTINCT manen.objectnaam) < 5<br>ORDER BY objectnaam;   |  |
|      |                                  |   | SELECT b.diameter AS diameter, f, count(*) as "cf", CAST(f*100.00/tot AS NUMERIC(5,2))AS rf,   |   |

|      |                                     |   |  |  |
|------|-------------------------------------|---|--|--|
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de volledige frequentietabel voor de diameters van de hemelobjecten (frequentie: hoeveel ojecten zijn er met de gegeven diameter, cumulatieve Frequentie, relatieve frequentie, Relatieve cumulatieve frequentie). Let op de datatypes en de precisie, gebruik CAST, rond niet af. Sorteer op diameter.  | CAST(100.00*count(*)/tot AS NUMERIC(5,2)) as "crf"<br>FROM (SELECT diameter, count(*) AS f<br>FROM hemelobjecten<br>GROUP BY diameter) AS b<br>INNER JOIN hemelobjecten bo ON (b.diameter >= bo.diameter),<br>(SELECT count(*) as tot FROM hemelobjecten) AS boe<br>GROUP BY b.diameter, f, tot<br>ORDER BY 1;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef alle niet-bezochte hemelobjecten, buiten het grootste hemellichaam.<br>Sorteer op diameter en objectnaam.  | SELECT objectnaam, afstand, diameter<br>FROM hemelobjecten<br><br>WHERE diameter < ANY (<br>SELECT diameter<br>FROM hemelobjecten rest<br>WHERE rest.objectnaam <> hemelobjecten.objectnaam<br>)<br><br>AND NOT EXISTS (<br>SELECT *<br>FROM bezoeken<br>WHERE bezoeken.objectnaam =<br>hemelobjecten.objectnaam<br>)<br>ORDER BY diameter, objectnaam;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de planeten en het aantal verschillende personen die hen bezocht hebben (met of zonder verblijf).<br>Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam, COUNT(DISTINCT<br>deelnames.klantnr) AS aantalbezoekers<br>FROM hemelobjecten<br>LEFT OUTER JOIN bezoeken ON<br>hemelobjecten.objectnaam = bezoeken.objectnaam<br>LEFT OUTER JOIN deelnames ON bezoeken.reisnr =<br>deelnames.reisnr<br>WHERE hemelobjecten.satellietvan = 'Zon'<br>GROUP BY hemelobjecten.objectnaam<br>ORDER BY hemelobjecten.objectnaam;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de klant die het meest op de maan is geweest (+het aantal).<br>Sorteer van voor naar achter.   | SELECT d.klantnr, count(b.objectnaam)<br>FROM (deelnames d INNER JOIN reizen r USING (reisnr))<br>INNER JOIN bezoeken b USING (reisnr)<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr<br>HAVING count(b.objectnaam) =<br>(SELECT max(c)<br>FROM (SELECT d.klantnr, count(b.objectnaam) AS c<br>FROM (deelnames d INNER JOIN reizen r USING (reisnr))<br>INNER JOIN bezoeken b<br>USING (reisnr)<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr) AS temp)<br>order by 1,2   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Maak een lijst met die mensen die meer dan 2 maal een reis ondernomen hebben waarin men geen enkele satelliet van Jupiter bezoekt !. Sorteer van voor naar achter.  | SELECT k.klantnr, (k. naam    k.vnaam) AS "Volledige<br>naam", count(d.reisnr) AS "Aantal Ondernomen Reizen"<br>FROM klanten k JOIN deelnames d USING (k.klantnr)<br>WHERE d.reisnr NOT IN<br>(SELECT b.reisnr<br>FROM bezoeken b NATURAL INNER JOIN hemelobjecten<br>o<br>WHERE o.satellietvan = 'Jupiter')<br>GROUP BY k.klantnr, k. naam    k.vnaam<br>HAVING count(d.reisnr)>2<br>order by 1,2,3   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef de gemiddelde afstanden tot hun planeet van alle satellieten van Saturnus, afgerond op 2 getallen na de komma  | SELECT round(AVG(S.afstand), 2) as "Gemiddelde<br>afstand"<br>FROM hemelobjecten P, hemelobjecten S<br>WHERE P.objectnaam = 'Saturnus' AND S.satellietvan =<br>P.objectnaam  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Welke planeten met meer dan 7 manen worden bezocht (met of zonder verblijf)?<br>Sorteer aflopend op basis van het aantal manen.   | SELECT planeten.objectnaam<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten satellieten ON<br>satellieten.satellietvan = planeten.objectnaam<br>INNER JOIN hemelobjecten sterren ON<br>planeten.satellietvan = sterren.objectnaam<br>INNER JOIN bezoeken ON planeten.objectnaam =<br>bezoeken.objectnaam<br>WHERE sterren.satellietvan IS NULL<br>GROUP BY planeten.objectnaam<br>HAVING COUNT(DISTINCT satellieten.objectnaam) > 7<br>ORDER BY COUNT(DISTINCT satellieten.objectnaam)<br>DESC;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag B | Geef voor elk hemelobject de minimale en maximale gemiddelde afstand tot zijn zon (de centrale ster in een sterrenstelsel) als je weet dat de kolom 'afstand' de gemiddelde afstand bevat tot het hemelobject waarond ze draaien. Met de grootte van het hemelobject hoeft je geen rekening te houden.<br>Sorteer op minimale afstand en op objectnaam. | SELECT hemelobjecten.objectnaam,<br>CASE<br>WHEN hemelobjecten.afstand IS NULL THEN 0<br>WHEN draaitrond.afstand IS NULL THEN<br>hemelobjecten.afstand<br>ELSE draaitrond.afstand + hemelobjecten.afstand<br>END AS maximale_afstand,<br>CASE<br>WHEN hemelobjecten.afstand IS NULL THEN 0<br>WHEN draaitrond.afstand IS NULL THEN<br>hemelobjecten.afstand<br>ELSE draaitrond.afstand - hemelobjecten.afstand<br>END AS minimale_afstand<br>FROM hemelobjecten<br>LEFT OUTER JOIN hemelobjecten draaitrond ON<br>hemelobjecten.satellietvan = draaitrond.objectnaam<br>ORDER BY minimale_afstand, objectnaam; |  |
| 2023 | Databanken:<br>DB2                  | Welke planeten met meer dan 7 manen worden bezocht (met of zonder verblijf)?<br>Sorteer aflopend op basis van het aantal manen.   | SELECT planeten.objectnaam<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten satellieten ON<br>satellieten.satellietvan = planeten.objectnaam<br>INNER JOIN bezoeken ON planeten.objectnaam =<br>bezoeken.objectnaam  |  |










|      |                                  |   |   |  |
|------|----------------------------------|---|---|--|
|      | Examenvraag B                    | <p>Let erop dat je planeten die meerdere keren bezocht worden, niet dubbel telt.</p>  | <p>WHERE planeten.satellietvan = 'Zon'<br/>GROUP BY planeten.objectnaam<br/>HAVING COUNT(DISTINCT satellieten.objectnaam) &gt; 7<br/>ORDER BY COUNT(DISTINCT satellieten.objectnaam) DESC;</p>  |  |
| 2023 | Databanken: DB2<br>Examenvraag B | <p>Geef per klant het totaal aantal reizen waaraan deze klant zal deelnemen en het langste bezoek dat deze klant zal maken aan een hemelobject, over alle reizen heen.<br/>Klanten zonder reizen of zonder bezoeken moeten ook voorkomen in het overzicht.<br/>Sorteer op klantnr.</p>  | <p>SELECT klanten.klantnr, COUNT(DISTINCT reizen.reisnr) AS aantal, MAX(bezoeken.verblijfsduur) AS langstebezoek<br/>FROM klanten<br/>LEFT OUTER JOIN deelnames ON klanten.klantnr = deelnames.klantnr<br/>LEFT OUTER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br/>LEFT OUTER JOIN bezoeken ON deelnames.reisnr = bezoeken.reisnr<br/>GROUP BY klanten.klantnr<br/>ORDER BY klanten.klantnr;</p> |  |
| 2023 | Databanken: DB2<br>Examenvraag B | <p>Geef enkel de 2 voorlaatste reizen terug. De positie van reizen wordt bepaalt door het reisnummer.<br/>Ter vergelijking, als je de getallen 1 t/m 10 neemt, dan is dit 8 en 9. Sorteer van voor naar achter.<br/>Gebruik enkel ISO sql.</p>  | <p>SELECT *<br/>from<br/>(select * from reizen order by reisnr desc offset 1 fetch first 2 rows only) as t<br/>order by 1,2,3,4;</p>  |  |
| 2023 | Databanken: DB2<br>Examenvraag C | <p>Welke van de volgende stellingen zijn JUIST?:</p> <p>A Relationale databanken zijn achterhaald.<br/>B PG_BENCH kan je gebruiken om de kostprijs van queries te berekenen<br/>C Een MongoDB installatie voldoet aan tegelijk aan C A en P<br/>D Een goeie PostgreSQL installatie voldoet tegelijk aan C A en P<br/>E Een standaard MySQL installatie voldoet niet tegelijk aan C A en P<br/>F PostgreSQL is een zuiver relationele databank<br/>G PGAdmin is een beperkte relationele databank<br/>H NoSQL databanken zijn altijd sneller dan relationele databanken<br/>I NoSQL voldoet aan meestal aan B, A, S en E<br/>J Een expliciete INNER JOIN geeft betere prestatie dan dezelfde conditie in de WHERE</p> <p>Antwoord bv ABC als je denkt dat de stellingen A,B en C juist zijn, er is geen feedback of het antwoord juist is. Het is normaal dat dit een foutmelding geeft</p>  |   |  |
| 2023 | Databanken: DB2<br>Examenvraag C | <p>Welke van de volgende stellingen zijn VERKEERD?:</p> <p>A Relationale databanken zijn achterhaald.<br/>B PG_BENCH kan je gebruiken om de kostprijs van queries te berekenen<br/>C Een MongoDB installatie voldoet aan tegelijk aan C A en P<br/>D Een goeie PostgreSQL installatie voldoet tegelijk aan C A en P<br/>E Een standaard MySQL installatie voldoet niet tegelijk aan C A en P<br/>F PostgreSQL is een zuiver relationele databank<br/>G PGAdmin is een beperkte relationele databank<br/>H NoSQL databanken zijn altijd sneller dan relationele databanken<br/>I NoSQL voldoet aan meestal aan B, A, S en E<br/>J Een expliciete INNER JOIN geeft betere prestatie dan dezelfde conditie in de WHERE</p> <p>Antwoord bv ABC als je denkt dat de stellingen A,B en C het antwoord zijn, er is geen feedback op je oplossing. Het is normaal dat dit een foutmelding geeft</p>   |   |  |
| 2023 | Databanken: DB2<br>Examenvraag C | <p>Welke van de volgende stellingen zijn JUIST?:</p> <p>A Relationale databanken zijn achterhaald.<br/>B PG_BENCH kan je gebruiken om de kostprijs van queries te berekenen<br/>C Een MongoDB installatie voldoet aan tegelijk aan C A en P<br/>D NoSQL databanken zijn altijd sneller dan relationele databanken<br/>E NoSQL voldoet aan meestal aan B, A, S en E<br/>F Een expliciete INNER JOIN geeft betere prestatie dan dezelfde conditie in de WHERE<br/>G Een goeie PostgreSQL installatie voldoet tegelijk aan C A en P<br/>H Een standaard MySQL installatie voldoet niet tegelijk aan C A en P<br/>I PostgreSQL is een zuiver relationele databank<br/>J PGAdmin is een beperkte relationele databank</p> <p>Antwoord bv ABC als je denkt dat de stellingen A,B en C juist zijn, er is geen feedback of het antwoord juist is. Het is normaal dat dit een foutmelding geeft.</p>   |   |  |
| 2023 | Databanken: DB2<br>Examenvraag D | <p>Toon voor alle planeten -die nog niet bezocht zijn- hun objectnaam en diameter als xml element, waarbij de objectnaam een metadata attriboot is van de diameter. Sorteer standaard op diameter, en vervolgens op objectnaam.</p> <p>Planeet: draait om een ster, niet om een andere planeet.<br/>Ster: staat op zich (bv de Zon), er zijn meerdere sterren in het sterrenstelsel</p> <p>(Gebruik geen LATERAL of CTE)t</p> <p>xlminfo uitvoer (de andere lijnen hebben een analoge opmaak, de uitvoer wordt dus volgens deze opmaak gevraagd):<br/>&lt;diameter objectnaam="Pluto"&gt;2324&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Mercurius"&gt;4878&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Venus"&gt;12104&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Aarde"&gt;12756&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Neptunus"&gt;50538&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Uranus"&gt;51118&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Saturnus"&gt;120536&lt;/diameter&gt;</p> | <p>select<br/>xmlelement(name diameter,<br/>xmlattributes(h.objectnaam),h.diameter)<br/>from<br/>(hemelobjecten h inner join hemelobjecten s on<br/>(h.satellietvan=s.objectnaam) and s.satellietvan is null) left<br/>outer join bezoeken b on (h.objectnaam=b.objectnaam)<br/>where<br/>b.reisnr is null<br/>order by<br/>h.diameter,h.objectnaam;</p>  |  |
|      |                                  | <p>Toon voor alle planeten -die al minstens 1 dag bezocht zijn- hun objectnaam en diameter als xml element, waarbij de objectnaam een metadata attriboot is van de diameter. Sorteer standaard op diameter, en vervolgens op objectnaam.</p>  |   |  |










|      |                                  |  |   |   |
|------|----------------------------------|--|---|---|
| 2023 | Databanken: DB2<br>Examenvraag D | <p>Planeet: draait om een ster, niet om een andere planeet.<br/>Ster: staat op zich (bv de Zon), er zijn meerdere sterren in het sterrenstelsel</p> <p>(Verblijfsduur: geeft het aantal dagen weer)</p> <p>(Gebruik geen LATERAL of CTE)</p> <p>xmlinfo uitvoer (de andere lijnen hebben een analoge opmaak, de uitvoer wordt dus volgens deze opmaak gevraagd):<br/>&lt;diameter objectnaam="Mars"&gt;6794&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Mars"&gt;6794&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Mars"&gt;6794&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Mars"&gt;6794&lt;/diameter&gt;</p>   | <pre>select xmlelement(name diameter, xmlattributes(h.objectnaam),h.diameter) from (hemelobjecten h inner join hemelobjecten s on (h.satellietvan=s.objectnaam) and s.satellietvan is null) inner join bezoeken b on (h.objectnaam=b.objectnaam) where b.verblijfsduur&gt;0 order by h.diameter,h.objectnaam;</pre>                           |    |
| 2023 | Databanken: DB2<br>Examenvraag D | <p>Toon alle planeten -die nog niet bezocht zijn- als json. Sorteer standaard op diameter, en vervolgens op objectnaam, satellietvan, afstand.</p> <p>Planeet: draait om een ster, niet om een andere planeet.<br/>Ster: staat op zich (bv de Zon), er zijn meerdere sterren in het sterrenstelsel</p> <p>(Gebruik geen LATERAL of CTE)</p> <p>json tip:* (zoals in count(h.*))</p>  | <pre>select to_json(h.*) from (hemelobjecten h inner join hemelobjecten s on (h.satellietvan=s.objectnaam) and s.satellietvan is null) left outer join bezoeken b on (h.objectnaam=b.objectnaam) where b.reisnr is null order by h.diameter,h.objectnaam, h.satellietvan,h.afstand;</pre>   |    |
| 2023 | Databanken: DB2<br>Examenvraag D | <p>Toon alle planeten -die al minstens 1 dag bezocht zijn- als json. Sorteer standaard op diameter, en vervolgens op objectnaam, satellietvan, afstand.</p> <p>Planeet: draait om een ster, niet om een andere planeet.<br/>Ster: staat op zich (bv de Zon), er zijn meerdere sterren in het sterrenstelsel</p> <p>(Verblijfsduur: geeft het aantal dagen weer)</p> <p>(Gebruik geen LATERAL of CTE)</p> <p>json tip:* (zoals in count(h.*))</p>   | <pre>select to_json(h.*) from (hemelobjecten h inner join hemelobjecten s on (h.satellietvan=s.objectnaam) and s.satellietvan is null) inner join bezoeken b on (h.objectnaam=b.objectnaam) where b.verblijfsduur&gt;0 order by h.diameter,h.objectnaam,h.satellietvan,h.afstand;</pre>   |    |
| 2023 | Databanken: DB2<br>Examenvraag E | <p>Welke van de volgende stellingen zijn JUIST?:</p> <p>A Relationele databanken zijn achterhaald en hebben geen toekomst.<br/>B PG_BENCH kan je gebruiken om de kostprijs van queries te berekenen<br/>C Een MongoDB installatie voldoet aan tegelijk aan C A en P<br/>D Een goeie PostgreSQL installatie voldoet tegelijk aan C A en P<br/>E Een standaard MySQL installatie voldoet niet tegelijk aan C A en P<br/>F PostgreSQL is een zuiver relationele databank<br/>G PGAdmin is een beperkte relationele databank<br/>H NoSQL databanken zijn altijd sneller dan relationele databanken<br/>I NoSQL voldoet meestal aan B, A, S en E<br/>J Een expliciete INNER JOIN geeft altijd betere prestatie dan dezelfde conditie in de WHERE</p> <p>Antwoord bv ABC als je denkt dat de stellingen A,B en C juist zijn, er is geen feedback of het antwoord juist is. Het is normaal dat dit een foutmelding geeft</p>  |   |  |
| 2023 | Databanken: DB2<br>Examenvraag E | <p>Welke van de volgende stellingen zijn JUIST?:</p> <p>A Relationele databanken zijn achterhaald en hebben geen toekomst.<br/>B PG_BENCH kan je gebruiken om de kostprijs van queries te berekenen<br/>C Een MongoDB installatie voldoet aan tegelijk aan C A en P<br/>D NoSQL databanken zijn altijd sneller dan relationele databanken<br/>E NoSQL voldoet meestal aan B, A, S en E<br/>F Een expliciete INNER JOIN geeft altijd betere prestatie dan dezelfde conditie in de WHERE<br/>G Een goeie PostgreSQL installatie voldoet tegelijk aan C A en P<br/>H Een standaard MySQL installatie voldoet niet tegelijk aan C A en P<br/>I PostgreSQL is een zuiver relationele databank<br/>J PGAdmin is een beperkte relationele databank</p> <p>Antwoord bv ABC als je denkt dat de stellingen A,B en C juist zijn, er is geen feedback of het antwoord juist is. Het is normaal dat dit een foutmelding geeft.</p>   |   |  |
| 2023 | Databanken: DB2<br>Examenvraag F | <p>Toon voor alle grote planeten -die nog niet bezocht zijn- hun objectnaam en diameter als xml element, waarbij de objectnaam een metadata attribuut is van de diameter.<br/>Grote planeten hebben hier een diameter groter dan 3000.<br/>Sorteer standaard op diameter, en vervolgens op objectnaam.</p> <p>Planeet: draait om een ster, niet om een andere planeet.<br/>Ster: staat op zich (bv de Zon), er zijn meerdere sterren in het sterrenstelsel</p> <p>(Gebruik geen LATERAL of CTE)</p> <p>xmlinfo uitvoer (de andere lijnen hebben een analoge opmaak, de uitvoer wordt dus volgens deze opmaak gevraagd):<br/>&lt;diameter objectnaam="Mercurius"&gt;4878&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Venus"&gt;12104&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Aarde"&gt;12756&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Neptunus"&gt;50538&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Uranus"&gt;51118&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Saturnus"&gt;120536&lt;/diameter&gt;</p> | <pre>select xmlelement(name diameter, xmlattributes(h.objectnaam),h.diameter) from (hemelobjecten h inner join hemelobjecten s on (h.satellietvan=s.objectnaam) and s.satellietvan is null) left outer join bezoeken b on (h.objectnaam=b.objectnaam) where b.reisnr is null and h.diameter &gt; 3000 order by h.diameter,h.objectnaam;</pre> |  |
|      |                                  | <p>Toon voor alle planeten -die al minstens 1 dag bezocht zijn- hun objectnaam en diameter als xml element, waarbij de objectnaam een metadata attribuut is van de diameter.<br/>Per bezoek dient er een uitvoerrij te zijn.</p>   |   |   |





|      |                                     |   |   |   |
|------|-------------------------------------|---|---|---|
| 2023 | Databanken:<br>DB2<br>Examenvraag F | <p>Sorteer standaard op diameter, en vervolgens op objectnaam.</p> <p>Planeet: draait om een ster, niet om een andere planeet.<br/>Ster: staat op zich (bv de Zon), er zijn meerdere sterren in het sterrenstelsel</p> <p>(Verblijfsduur: geeft het aantal dagen weer)</p> <p>(Gebruik geen LATERAL of CTE)</p> <p>xlminfo uitvoer (de andere lijnen hebben een analoge opmaak, de uitvoer wordt dus volgens deze opmaak gevraagd):<br/>&lt;diameter objectnaam="Mars"&gt;6794&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Mars"&gt;6794&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Mars"&gt;6794&lt;/diameter&gt;<br/>&lt;diameter objectnaam="Mars"&gt;6794&lt;/diameter&gt;</p>   | <pre>select xmlelement(name diameter, xmldata(h.objectnaam),h.diameter) from (hemelobjecten h inner join hemelobjecten s on (h.satellietvan=s.objectnaam) and s.satellietvan is null) inner join bezoeken b on (h.objectnaam=b.objectnaam) where b.verblijfsduur&gt;0 order by h.diameter,h.objectnaam;</pre> |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag G | <p>Geef voor elk bestuurslid de twee kleinste boetes. Sorteer op spelersnr, bedrag, naam, voorletters.</p> <p>(Elk bestuurslid: actief of niet)<br/>(Twee kleinste boetes: indien deze bestaan)</p> <p>Gebruik geen GTE/CTE.</p>  | <pre>select s.spelersnr,s.naam,s.voorletters,hs.bedrag from spelers s left join lateral ( select * from boetes f where f.spelersnr=s.spelersnr order by bedrag asc fetch first 2 rows only) hs on true where s.spelersnr in ( select spelersnr from bestuursleden) order by 1,4,2,3;</pre>                    |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag G | <p>Geef voor elke speler van team 1 de twee grootste boetes. Sorteer op spelersnr, bedrag, naam, voorletters.</p> <p>(van team 1: deze speler heeft een wedstrijd gespeeld voor team 1)<br/>(Twee grootste boetes: indien deze bestaan, andere boetes worden niet getoond, dus maximaal 2 per speler)</p> <p>Gebruik geen GTE/CTE.</p>  | <pre>select s.spelersnr,s.naam,s.voorletters,hs.bedrag from spelers s left join lateral ( select * from boetes f where f.spelersnr=s.spelersnr order by bedrag desc fetch first 2 rows only) hs on true where s.spelersnr in ( select spelersnr from wedstrijden w where w.teamnr=1) order by 1,4,2,3;</pre>  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag G | <p>Geef voor elke niet-aanvoerder de twee kleinste boetes. Sorteer op spelersnr, bedrag, naam, voorletters.</p> <p>(Elk niet-aanvoerder: spelers die geen kapitein/aanvoerder zijn van een team)<br/>(Twee kleinste boetes: indien deze bestaan)</p> <p>Gebruik geen GTE/CTE.</p>   | <pre>select s.spelersnr,s.naam,s.voorletters,hs.bedrag from spelers s left join lateral ( select * from boetes f where f.spelersnr=s.spelersnr order by bedrag asc fetch first 2 rows only) hs on true where s.spelersnr not in ( select spelersnr from teams) order by 1,4,2,3;</pre>                        |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag G | <p>Geef voor elke klant zijn reizen en hoelang het geleden is dat deze op reis is gegaan (sinds de vorige reis dus).</p> <p>Sorteer op 1,3 desc,2,4,6;</p> <p>Gebruik geen CTE/GTE</p> <p>Uitvoer hint:<br/>klantnr   reisnr   vertrekdatum   reisduur   prijs   hoelang_geleden<br/>-----+-----+-----+-----+-----<br/>121   33   2020-10-12   11   2.65   131<br/>121   32   2020-06-03   390   17.50   1<br/>121   31   2020-06-02   10   2.50  <br/>122   34   2021-01-10   1380   75.00   222<br/>122   31   2020-06-02   10   2.50  <br/>123   32   2020-06-03   390   17.50   1<br/>123   31   2020-06-02   10   2.50  <br/>124   32   2020-06-03   390   17.50   1<br/>124   31   2020-06-02   10   2.50  <br/>125   34   2021-01-10   1380   75.00   90<br/>125   33   2020-10-12   11   2.65  <br/>126   34   2021-01-10   1380   75.00   90<br/>126   33   2020-10-12   11   2.65  <br/>(13 rows)</p> | <pre>select h.klantnr,s.*,s.vertrekdatum - lag(s.vertrekdatum) over(partition by h.klantnr order by s.vertrekdatum asc) as hoelang_geleden from deelnames h left outer join reizen s on (h.reisnr=s.reisnr) order by 1,3 Desc,2,4,6;</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag H | <p>Geef voor elk hemelobject zijn satellieten en cumulatief de afstanden (van de satellieten tot hun hemelobject) wanneer je de satellieten van klein naar groot sorteert.</p> <p>Sorteer op kolom1, diameter aflopend, kolom2, kolom3, kolom4, kolom6.</p> <p>Gebruik geen CTE/GTE</p>   | <pre>select h.objectnaam as elk_hemelobject,s.*,sum(s.afstand) over(partition by h.objectnaam order by s.diameter) from hemelobjecten h left outer join hemelobjecten s on (h.objectnaam=s.satellietvan) order by 1, s.diameter desc,2,3,4,6;</pre>   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag H | <p>Geeft voor elk hemelobject de bezoeken en cumulatief de verblijfsduur op dit hemelobject, gesorteerd volgens reisnr en volgnr.</p> <p>Sorteer op 1,2,4,3,5,6.</p> <p>Gebruik geen CTE/GTE</p>  | <pre>select h.objectnaam as elk_hemelobject,s.*,sum(s.verblijfsduur) over(partition by h.objectnaam order by s.reisnr,s.volgnr) from hemelobjecten h left outer join bezoeken s on (h.objectnaam=s.objectnaam) order by 1, 2,4,3,5,6;</pre>   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag H | <p>Geef voor alle reizen met deelnames hun klanten en het verschil in leeftijd wanneer je de klanten van oud naar jong sorteert.</p> <p>Sorteer op 1,5,2,3,4,6</p> <p>Gebruik geen CTE/GTE</p>  | <pre>select h.reisnr,s.*,s.geboortedatum - lag(s.geboortedatum) over(partition by h.reisnr order by s.geboortedatum) as leeftijdsverschil from deelnames h left outer join klanten s on (h.klantnr=s.klantnr) order by 1, 5,2,3,4,6;</pre>  |  |

|      |                                  |   |   |   |
|------|----------------------------------|---|---|---|
| 2023 | Databanken: DB2<br>Examenvraag H | <p>De hoeveelste klant ben je op een reis?</p> <p>Geef per reis voor elke klant de hoeveelste klant deze is (gesorteerd volgens klantnr) voor die reis. Reizen zonder klanten hebben we niet nodig.</p> <p>Sorteer 1,2,3</p>  | <p>select s.*, rank() over(partition by h.reisnr order by s.klantnr) as groep from reizen h inner join deelnames s on (h.reisnr=s.reisnr) order by 1,2,3;</p>   |    |
| 2023 | Databanken: Exameninfo Augustus  | <p>Gebruik geen CTE/GTE</p> <p>Lees eerst dit alvorens aan het examen te beginnen.</p> <p>Het is een open boek examen via pc, u mag dus verschillende papieren en elektronische bronnen gebruiken. Belangrijk is dat dit gebruik 'niet-interactief' is. Interactief gebruik is niet toegelaten en wordt beschouwd als fraude. Een voorbeeld van interactief gebruik is aan een derde -direct of indirect- hulp te vragen.</p> <p>De maximumduur van het examen is 2 uur 36 minuten. Het examen zelf is voorzien om opgelost te kunnen worden op 1 uur 30 minuten. Er wordt geen rekening gehouden met antwoorden die te laat of in de verkeerde vorm worden ingediend.</p> <p>Enkel antwoorden die op de gevraagde manier worden ingediend tellen; er wordt geen rekening gehouden met andere vormen van indiening. Je dient dus op de juiste manier in te dienen. Bv een query vraag in sqldropbox dient via sqldropbox te worden ingediend en niet anders.</p> <p>Dit examen is persoonlijk per student, het gedrag of de behandeling omtrent het naleven van deze regels van andere studenten -die zich mogelijk niet aan deze regels houden- geven u niet het recht de hier gegeven regels te overtreden. U werkt strikt individueel Het examenreglement is van toepassing. Overtredingen van de hier en in het OER gegeven regels worden beschouwd als fraude.</p> <p>Bij de minste onduidelijkheid van deze regels brengt u onmiddellijk de beschikbare lector hiervan op de hoogte. De algemene richtlijnen staan op: <a href="https://intranet.ucll.be/nl/student/studeren-aan-ucll/pba-de-toegepaste-informatica-proximus/examen-en-evaluatie/richtlijnen-voor-studenten-bij-online-evalueren">https://intranet.ucll.be/nl/student/studeren-aan-ucll/pba-de-toegepaste-informatica-proximus/examen-en-evaluatie/richtlijnen-voor-studenten-bij-online-evalueren</a> Gelieve deze op voorhand door te nemen.</p> <p>Het is jouw taak om te zorgen dat benodigde hardware, software en internet werken. De vak specifieke invulling staat op: <a href="https://projektwerk.ucll.be/projects/db2/wiki/Alternatieve_examinatie">https://projektwerk.ucll.be/projects/db2/wiki/Alternatieve_examinatie</a> Gelieve deze op voorhand door te nemen.</p> <p>Hierbij stemt u in met deze regels en geeft u te kennen deze zonder misverstanden begrepen te hebben, dit doet u door eerst bij aanvang van het examen u naam, handtekening en de woorden 'gelezen en goedgekeurd' te onderschrijven alvorens verder te werken. Dit doe je door letterlijk " select 'gelezen en goedgekeurd' " in te geven in sqldropbox.</p> | <p>select 'gelezen en goedgekeurd';</p>   |    |
| 2023 | Databanken: DB2<br>Examenvraag i | <p>Geef alle bestuursleden die geboren zijn voor 1970-08-05 en hun staat van dienst. Bekijk de output om de juiste formatering te zien. gebruik een combinatie van CASE, de    functie en datum opmaak. Sorteer op naam en begin datum</p>  | <p>SELECT spelers.voorletters, spelers.naam, bestuursleden.functie, 'actief: '    CASE WHEN bestuursleden.eind_datum IS NULL THEN 'sinds '    to_char(bestuursleden.begin_datum, 'DD/MM/YYYY') ELSE 'van '    to_char(bestuursleden.begin_datum, 'DD/MM')    ' tot '    to_char(bestuursleden.eind_datum, 'DD/MM/YYYY') END AS periode FROM bestuursleden, spelers WHERE bestuursleden.spelersnr = spelers.spelersnr AND spelers.geb_datum &lt; '1970-08-05' ORDER BY spelers.naam, bestuursleden.begin_datum</p> |  |
| 2023 | Databanken: DB2<br>Examenvraag i | <p>Geef een overzicht van alle spelers met een bondsnummer. Sorteer als volgt:</p> <ul style="list-style-type: none"> <li>- Eerst moeten alle spelers uit zoetermeer getoond worden, deze sorteer je op naam. (spelers met dezelfde naam uit zoetermeer worden gesorteerd op bondsnr)</li> <li>- Daarna de andere spelers, gesorteerd op bondsnr.</li> </ul> <p>Tip: je kan CASE ook gebruiken in de ORDER BY component (zet dan '0')</p>   | <p>SELECT spelersnr, naam, plaats, bondsnr FROM spelers WHERE bondsnr IS NOT NULL ORDER BY CASE WHEN plaats = 'Zoetermeer' THEN '0' ELSE bondsnr END, naam, bondsnr;</p>  |  |
| 2023 | Databanken: DB2<br>Examenvraag i | <p>Geef voor alle bestuursleden een overzicht met hun naam, functie, de leeftijd die ze hadden op het moment van de start van hun functie en de leeftijd die ze hadden op het moment van het einde van hun functie.</p> <p>Tip: gebruik de postgresqlfuncties AGE en EXTRACT om het aantal jaar te krijgen. En CAST(...) as CHAR(2).</p> <p>(Dit is een moeilijke oefening)</p>   | <p>SELECT spelers.spelersnr, spelers.naam, bestuursleden.functie, CAST(EXTRACT(YEAR FROM AGE(begin_datum, geb_datum)) AS CHAR(2))    ' jaar' AS leeftijd_begin, CASE WHEN eind_datum IS NULL THEN 'nog bezig' ELSE CAST(EXTRACT(YEAR FROM AGE(eind_datum, geb_datum)) AS CHAR(2))    ' jaar' END AS leeftijd_einde FROM spelers, bestuursleden WHERE spelers.spelersnr = bestuursleden.spelersnr;</p>   |  |
| 2023 | Databanken: DB2<br>Examenvraag i | <p>Geef voor elke mannelijke speler wiens naam minstens 2 keer de letter 'e' bevat een lijst van de functies die hij op dit moment uitoefent. Ook mannelijke spelers zonder huidige functie moeten getoond worden. Sorteer op spelersnr.</p>  | <p>select naam, geslacht, functie from spelers left outer join bestuursleden on spelers.spelersnr = bestuursleden.spelersnr and bestuursleden.eind_datum is null where geslacht = 'M' and naam like '%e%e%' ORDER BY spelers.spelersnr;</p>   |  |
| 2023 | Databanken: DB2<br>Examenvraag i | <p>Sorteer de teams in functie van het aantal verloren wedstrijden (oplopend). Als het aantal verloren wedstrijden gelijk is, sorteer je op het totaal aantal gewonnen sets (aflopend) en op divisienaam (oplopend). Een wedstrijd die verloren werd met 3 - 0 telt niet mee in deze</p>  | <p>SELECT teams.divisie, COUNT(*) AS aantal FROM wedstrijden, teams WHERE wedstrijden.teamnr = teams.teamnr AND wedstrijden.verloren &gt; wedstrijden.gewonnen AND NOT (wedstrijden.verloren = 3 AND wedstrijden.gewonnen = 0)</p>  |  |

|      |                                  |  |  |   |
|------|----------------------------------|--|--|---|
|      |                                  | output.<br>Geef het aantal wedstrijden dat voldoet aan bovenstaande voorwaarde mee in de output.   | GROUP BY teams.divisie<br>ORDER BY aantal, SUM(wedstrijden.gewonnen) DESC, teams.divisie;  |   |
| 2023 | Databanken: DB2<br>Examenvraag i | Geef een lijst met de spelers die ooit bestuurslid zijn geweest (of nog steeds zijn) en niet in Den Haag of Zoetermeer wonen. Bijkomend mag deze speler maximaal 2 keer in het bestuur van de club gezeteld hebben.<br>De lijst moet aflopend gesorteerd worden op het aantal maal dat de betreffende speler in het bestuur zetelde. Mensen met hetzelfde aantal keren moeten oplopend gesorteerd worden op basis van hun spelersnr. | SELECT spelers.naam, COUNT(*) AS aantal<br>FROM spelers, bestuursleden<br>WHERE bestuursleden.spelersnr = spelers.spelersnr<br>AND plaats NOT IN ('Den Haag', 'Zoetermeer')<br>GROUP BY spelers.spelersnr, spelers.naam<br>HAVING COUNT(*) <= 2<br>ORDER BY aantal DESC, spelers.spelersnr;  |    |
| 2023 | Databanken: DB2<br>Examenvraag i | Geef het gemiddeld aantal gewonnen en verloren sets per geboortejaar.<br>Sorteer op geboortejaar.  | SELECT EXTRACT(YEAR FROM geb_datum) AS geboortejaar, AVG(gewonnen) AS gewonnen, AVG(verloren) AS verloren<br>FROM wedstrijden, spelers<br>WHERE wedstrijden.spelersnr = spelers.spelersnr<br>GROUP BY EXTRACT(YEAR FROM geb_datum)<br>ORDER BY geboortejaar;   |    |
| 2023 | Databanken: DB2<br>Examenvraag i | Geef spelers die in het jaar dat ze lid geworden zijn van de club reeds een boete van meer dan 50 euro gekregen hebben en de som van al deze boetes groter of gelijk is aan 100 euro.<br>Geef buiten de voorletters en de naam van de speler ook het aantal boetes die aan bovenstaande voorwaarden voldoen.<br>Sorteer op spelersnr.  | SELECT spelers.voorletters, spelers.naam, COUNT(*) AS aantalboetes<br>FROM spelers, boetes<br>WHERE spelers.spelersnr = boetes.spelersnr<br>AND boetes.bedrag > 50<br>AND EXTRACT(YEAR FROM boetes.datum) = spelers.jaartoe<br>GROUP BY spelers.spelersnr, spelers.voorletters, spelers.naam<br>HAVING SUM(bedrag) >= 100<br>ORDER BY spelers.spelersnr;   |    |
| 2023 | Databanken: DB2<br>Examenvraag i | Geef voor alle huidige bestuursleden hun functie en de lijst van boetes die voor hen werd betaald.<br>Omdat je dit wil vergelijken met de boetebedragen die betaald werden voor spelers die niet in het bestuur zitten, wil je deze boetebedragen ook opnemen in de tweede kolom van je resultaat. Sorteer je antwoord eerst op functie en daarna op het boetebedrag.  | SELECT functie, bedrag<br>FROM boetes BT FULL OUTER JOIN bestuursleden B<br>ON (BT.spelersnr = B.spelersnr<br>AND eind_datum is null)<br>ORDER BY 1,2;   |    |
| 2023 | Databanken: DB2<br>Examenvraag i | Geef per team de verloren wedstrijden. Zorg dat teams zonder verloren wedstrijden ook in de output verschijnen.<br>Duid per wedstrijd aan of het om een actief bestuurslid gaat.<br>Sorteer op divisie en wedstrijdnummer.   | SELECT teams.teamnr, teams.divisie, wedstrijden.wedstrijdnr, wedstrijden.spelersnr, CASE WHEN bestuursleden.spelersnr IS NULL THEN '-' ELSE 'actief' END AS bestuurslid<br>FROM teams<br>LEFT OUTER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr AND wedstrijden.verloren > wedstrijden.gewonnen<br>LEFT OUTER JOIN bestuursleden ON wedstrijden.spelersnr = bestuursleden.spelersnr AND bestuursleden.eind_datum IS NULL<br>ORDER BY divisie, wedstrijden.wedstrijdnr; |   |
| 2023 | Databanken: DB2<br>Examenvraag i | Geef voor de actieve bestuursleden zonder boete hun laatste gespeelde wedstrijd (die met het hoogste wedstrijdnummer).<br>Sorteer aflopend op spelersnr.   | SELECT bestuursleden.spelersnr, MAX(wedstrijden.wedstrijdnr) AS laatstewedstrijd<br>FROM bestuursleden<br>INNER JOIN wedstrijden ON bestuursleden.spelersnr = wedstrijden.spelersnr AND bestuursleden.eind_datum IS NULL<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr = boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY bestuursleden.spelersnr<br>ORDER BY spelersnr DESC;   |  |
| 2023 | Databanken: DB2<br>Examenvraag i | Geef het gemiddeld boetebedrag per speler, afgerond op twee cijfers na de komma. spelers zonder boete moeten bovenaan staan met 'geen boetes'<br>Sorteer daarna op spelersnaam en gemiddeld boetebedrag.<br>Om een waarde om te zetten naar een ander datatype, kan je ofwel CAST() gebruiken ofwel de TO_CHAR() met 'FM999.00' als opmaakmasker.  | SELECT spelers.naam, CASE WHEN AVG(boetes.bedrag) IS NULL THEN 'geen boetes' ELSE CAST(ROUND(AVG(boetes.bedrag), 2) AS varchar(8)) END AS gemiddeld<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr = boetes.spelersnr<br>GROUP BY spelers.spelersnr, spelers.naam<br>ORDER BY CASE WHEN AVG(boetes.bedrag) IS NULL THEN 0 ELSE 1 END, spelers.naam, AVG(boetes.bedrag);  |  |
| 2023 | Databanken: DB2<br>Examenvraag i | Geef per team de leeftijd van de aanvoerder (tip: postgresql heeft een AGE() functie) en het aantal verschillende spelers dat voor dit team gespeeld heeft.<br>Alleen teams waarvoor wedstrijden zijn gespeeld en die een aanvoerder hebben, moeten vermeld worden.<br>Sorteer op leeftijd en daarna op aantal verschillende spelers en daarna op teamnr.  | SELECT teams.teamnr, CAST(EXTRACT(year from AGE(NOW(), spelers.geb_datum)) AS varchar(2))    ' jaar' AS leeftijd, COUNT(DISTINCT wedstrijden.spelersnr) AS aantalspelers<br>FROM teams<br>INNER JOIN spelers ON teams.spelersnr = spelers.spelersnr<br>INNER JOIN wedstrijden ON teams.teamnr = wedstrijden.teamnr<br>GROUP BY teams.teamnr, spelers.geb_datum<br>ORDER BY leeftijd, aantalspelers, teamnr;  |  |
| 2023 | Databanken: DB2<br>Examenvraag i | Geef alle spelers die meer wedstrijden gespeeld hebben dan het aantal wedstrijden dat de huidige voorzitter heeft verloren. De huidige voorzitter komt zelf niet in de lijst voor. Gebruik geen subqueries.<br>Sorteer op spelersnr.   | SELECT spelers.spelersnr<br>FROM spelers<br>INNER JOIN wedstrijden ON spelers.spelersnr = wedstrijden.spelersnr, bestuursleden<br>INNER JOIN wedstrijden voorzitterswedstrijden ON bestuursleden.spelersnr = voorzitterswedstrijden.spelersnr<br>AND voorzitterswedstrijden.gewonnen < voorzitterswedstrijden.verloren<br>WHERE bestuursleden.eind_datum IS NULL AND bestuursleden.functie = 'Voorzitter' AND  |  |

|      |                                     |  |   |   |
|------|-------------------------------------|--|---|---|
|      |                                     |  | bestuursleden.spelersnr <> spelers.spelersnr<br>GROUP BY spelers.spelersnr<br>HAVING COUNT(DISTINCT wedstrijden.wedstrijdnr) ><br>COUNT(DISTINCT voorzitterswedstrijden.wedstrijdnr)<br>ORDER BY spelers.spelersnr;   |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef voor elke mannelijke speler wiens naam minstens 2 keer de letter 'e' bevat zijn functie die hij op dit moment uitoefent, als die er op dit moment één heeft.<br>Sorteer op naam en functie.   | select naam, geslacht, functie<br>from spelers left outer join bestuursleden<br>on spelers.spelersnr = bestuursleden.spelersnr<br>and bestuursleden.eind_datum is null<br>where geslacht = 'M'<br>and naam like '%e%e%'<br>ORDER BY naam, functie;  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef per team de verloren wedstrijden. Zorg dat teams zonder verloren wedstrijden ook in de output verschijnen.<br>Duid per wedstrijd aan of het om een actief bestuurslid gaat.<br>Sorteer op divisie en wedstrijdnummer.   | SELECT teams.teamnr, teams.divisie,<br>wedstrijden.wedstrijdnr, wedstrijden.spelersnr,<br>CASE<br>WHEN bestuursleden.spelersnr IS NULL THEN '-'<br>ELSE 'actief'<br>END AS bestuurslid<br>FROM teams<br>LEFT OUTER JOIN wedstrijden ON teams.teamnr =<br>wedstrijden.teamnr AND wedstrijden.verloren ><br>wedstrijden.gewonnen<br>LEFT OUTER JOIN bestuursleden ON<br>wedstrijden.spelersnr = bestuursleden.spelersnr AND<br>bestuursleden.eind_datum IS NULL<br>ORDER BY divisie, wedstrijden.wedstrijdnr;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef het gemiddeld boetebedrag per speler, afgerond op twee cijfers na de komma. spelers zonder boete moeten bovenaan staan met 'geen boetes'<br>Sorteer daarna op spelersnaam en vervolgens op gemiddeld boetebedrag.<br>Gebruik als datatype van de 2de kolom varchar(8) voor spelers die wel boetes hebben.   | SELECT spelers.naam,<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 'geen boetes'<br>ELSE CAST(ROUND(AVG(boetes.bedrag), 2) AS<br>varchar(8))<br>END<br>AS gemiddeld<br>FROM spelers<br>LEFT OUTER JOIN boetes ON spelers.spelersnr =<br>boetes.spelersnr<br>GROUP BY spelers.spelersnr, spelers.naam<br>ORDER BY<br>CASE<br>WHEN AVG(boetes.bedrag) IS NULL THEN 0<br>ELSE 1<br>END, spelers.naam, AVG(boetes.bedrag);  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef per team de leeftijd van de aanvoerder (tip: postgresql heeft een AGE() functie) en het aantal verschillende spelers dat voor dit team gespeeld heeft. Alleen teams waarvoor wedstrijden zijn gespeeld en die een aanvoerder hebben, moeten vermeld worden.<br>Sorteer op leeftijd en daarna op aantal verschillende spelers en daarna op teamnr. | SELECT teams.teamnr, CAST(EXTRACT(year from<br>AGE(NOW(), spelers.geb_datum)) AS varchar(2))    ' jaar'<br>AS leeftijd, COUNT(DISTINCT wedstrijden.spelersnr) AS<br>aantalspelers<br>FROM teams<br>INNER JOIN spelers ON teams.spelersnr =<br>spelers.spelersnr<br>INNER JOIN wedstrijden ON teams.teamnr =<br>wedstrijden.teamnr<br>GROUP BY teams.teamnr, spelers.geb_datum<br>ORDER BY leeftijd, aantalspelers, teamnr;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef per team het hoogste wedstrijdnummer van een wedstrijd, gespeeld door een bestuurslid (actief en niet meer actief) die geen boete heeft gekregen.<br>Sorteer op teamnr.   | SELECT teams.teamnr, MAX(wedstrijden.wedstrijdnr) AS<br>laatstewedstrijd<br>FROM teams<br>INNER JOIN wedstrijden on teams.teamnr =<br>wedstrijden.teamnr<br>INNER JOIN bestuursleden ON wedstrijden.spelersnr =<br>bestuursleden.spelersnr<br>LEFT OUTER JOIN boetes ON bestuursleden.spelersnr =<br>boetes.spelersnr<br>WHERE boetes.spelersnr IS NULL<br>GROUP BY teams.teamnr<br>ORDER BY teams.teamnr;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef alle spelers die meer wedstrijden gespeeld hebben dan het aantal wedstrijden dat de huidige voorzitter heeft verloren. De huidige voorzitter komt zelf niet in de lijst voor. Gebruik geen subqueries.<br>Sorteer op spelersnr.   | SELECT spelers.spelersnr<br>FROM spelers<br>INNER JOIN wedstrijden ON spelers.spelersnr =<br>wedstrijden.spelersnr,<br>bestuursleden<br>INNER JOIN wedstrijden voorzitterswedstrijden ON<br>bestuursleden.spelersnr = voorzitterswedstrijden.spelersnr<br>AND voorzitterswedstrijden.gewonnen <<br>voorzitterswedstrijden.verloren<br>WHERE bestuursleden.eind_datum IS NULL AND<br>bestuursleden.functie = 'Voorzitter' AND<br>bestuursleden.spelersnr <> spelers.spelersnr<br>GROUP BY spelers.spelersnr<br>HAVING COUNT(DISTINCT wedstrijden.wedstrijdnr) ><br>COUNT(DISTINCT voorzitterswedstrijden.wedstrijdnr)<br>ORDER BY spelers.spelersnr; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef voor elke speler die een wedstrijd heeft gespeeld het spelersnr en het totaal aantal boetes. Spelers die een wedstrijd gespeeld hebben, maar geen boetes hebben, moeten ook getoond worden.<br>Sorteer op het aantal boetes en op spelersnr;  | SELECT DISTINCT w.spelersnr, aantalboetes<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes<br>RIGHT OUTER JOIN wedstrijden w USING (spelersnr)<br>ORDER BY aantalboetes   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef voor alle spelers die geen penningmeester zijn of zijn geweest alle gewonnen wedstrijden, gesorteerd op wedstrijdnummer.  | SELECT spelersnr, wedstrijdnr<br>FROM wedstrijden<br>WHERE spelersnr NOT IN (<br>SELECT spelersnr<br>FROM bestuursleden<br>WHERE functie = 'Penningmeester')<br>AND gewonnen > verloren<br>ORDER BY wedstrijdnr;  |  |
| 2023 | Databanken:<br>DB2                  | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer op   | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe-<br>avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers  |  |









|      |                                     |   |   |   |
|------|-------------------------------------|---|---|---|
|      | Examenvraag i                       | spelersnr. Toon 3 getallen na de komma, zet het verschil om naar het numeric type met precisie van 5 en een schaal van 3.   | GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1   |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Je kan per speler berekenen hoeveel boetes die speler heeft gehad en wat het totaalbedrag per speler is. Pas nu deze query aan zodat per verschillend aantal boetes wordt getoond hoe vaak dit aantal boetes voorkwam.Sorteer eerst op de eerste kolom en daarna op de tweede kolom.  | SELECT a, count(a)<br>FROM (SELECT count(bedrag) AS a<br>FROM boetes<br>GROUP BY spelersnr) b<br>GROUP BY a<br>ORDER BY 1,2   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef voor elke aanvoerder het spelersnr, de naam en het aantal boetes dat voor hem of haar betaald is en het aantal teams dat hij of zij aanvoert. Aanvoerders zonder boetes mogen niet getoond worden. Sorteer, beginnend bij de eerste kolom, eindigend bij de laatste kolom.   | SELECT s.spelersnr, s.naam, s.voorletters, aantalboetes, aantalteams<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes,<br>(SELECT spelersnr, count(*) AS aantalteams<br>FROM teams<br>GROUP BY spelersnr) AS aantal_teams,<br>spelers s<br>WHERE s.spelersnr=aantal_boetes.spelersnr<br>AND s.spelersnr=aantal_teams.spelersnr<br>ORDER BY 1,2,3,4,5  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer op spelersnr. Zet het berekende verschil om naar het datatype numeric met precisie 5 en schaal 3.   | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe-<br>avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef een lijst van alle spelers (spelersnr en woonplaats) die met minstens twee in dezelfde plaats wonen. Sorteer aflopend op woonplaats, daarna op spelersnr.  | SELECT spelersnr, plaats<br>FROM spelers<br>WHERE plaats IN (<br>SELECT plaats<br>FROM spelers<br>GROUP BY plaats<br>HAVING COUNT(*) >= 2<br>)<br>ORDER BY plaats DESC, spelersnr;  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef de spelersgegevens van de speler(s) met het hoogste bedrag (voor één boete, niet het totaalbedrag). Als twee spelers een even hoge boete gehad hebben, moeten beide spelers getoond worden (LIMIT is dus geen optie). Sorteer alfabetisch op naam en voorletters.  | SELECT spelers.spelersnr, voorletters, naam<br>FROM spelers<br>INNER JOIN boetes using (spelersnr)<br>WHERE boetes.bedrag = (<br>SELECT MAX(bedrag)<br>FROM boetes<br>)<br>ORDER BY naam, voorletters   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | We willen een statistiek van hoeveel wedstrijden de spelers winnen.<br>Geef een lijst met aantal gewonnen wedstrijden en het aantal spelers dat dit aantal wedstrijden gewonnen heeft.<br>Dus bv als er vier spelers zijn die elk drie wedstrijden hebben gewonnen, dan is de output: aantal_gewonnen: 3, aantal_spelers: 4. Dit graag voor alle aantallen gewonnen wedstrijden en alle spelers.<br>Sorteer op aantal gewonnen wedstrijden. | SELECT aantal_gewonnen, count(spelersnr) AS<br>aantal_spelers<br>FROM (<br>SELECT spelersnr, count(*) AS aantal_gewonnen<br>FROM wedstrijden<br>WHERE gewonnen > verloren<br>GROUP BY spelersnr) AS gewonnen<br>GROUP BY aantal_gewonnen<br>ORDER BY aantal_gewonnen;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef het totaal aantal boetes, het totale boetebedrag, het minimum en het maximum boetebedrag dat door onze club betaald werd. Let er hierbij op dat er gehele getallen worden getoond (rond af indien nodig). Sorteer van voor naar achter, oplopend.  | SELECT count(*) AS Aantal_boetes, round(sum(bedrag))<br>AS Totaal_bedrag, round(min(bedrag)) as Minimum,<br>round(max(bedrag)) as Maximum<br>FROM boetes<br>ORDER BY 1,2,3,4;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef voor elke aanvoerder het spelersnr, de naam en het aantal boetes dat voor hem of haar betaald is en het aantal teams dat hij of zij aanvoert. Toon enkel aanvoerders die boetes gekregen hebben. Sorteer van voor naar achter, oplopend.   | SELECT s.spelersnr, s.naam, s.voorletters, aantalboetes, aantalteams<br>FROM (SELECT spelersnr, count(*) AS aantalboetes<br>FROM boetes<br>GROUP BY spelersnr) AS aantal_boetes,<br>(SELECT spelersnr, count(*) AS aantalteams<br>FROM teams<br>GROUP BY spelersnr) AS aantal_teams,<br>spelers s<br>WHERE s.spelersnr=aantal_boetes.spelersnr<br>AND s.spelersnr=aantal_teams.spelersnr<br>ORDER BY 1,2,3,4,5; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef alle spelers voor wie meer boetes zijn betaald dan dat ze wedstrijden hebben gespeeld. Zorg dat spelers zonder wedstrijd ook getoond worden.<br>Sorteer van voor naar achter, oplopend.  | SELECT s.naam, s.voorletters, s.geb_datum<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr, s.naam, s.voorletters, s.geb_datum<br>HAVING count(*) ><br>(<br>SELECT count(*)<br>FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr)<br>ORDER BY 1,2,3  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef voor elke speler die ooit een boete heeft betaald, de hoogste boete weer en hoelang het geleden is dat deze boete werd betaald. Sorteer van groot naar klein op bedrag en daarna omgekeerd op "leeftijd.." v van de boete.   | SELECT sub.spelersnr, sub.bedrag, age(b.datum)<br>FROM (<br>SELECT s.spelersnr, max(b.bedrag) AS bedrag<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr)<br>INNER JOIN boetes b USING (spelersnr)<br>WHERE sub.bedrag = b.bedrag<br>ORDER BY 2 DESC, 3  |  |
| 2023 | Databanken:<br>DB2                  | Welke spelers hebben voor alle teams gespeeld uit de teamstabel ?<br>(= voor welke speler bestaat er geen enkel team waar de  | SELECT s.spelersnr<br>FROM spelers s<br>WHERE NOT EXISTS<br>(<br>SELECT '1'<br>FROM teams t<br>WHERE NOT EXISTS   |  |



|      |                                     |   |   |   |
|------|-------------------------------------|---|---|---|
|      | Examenvraag i                       | betreffende speler nooit voor gespeeld heeft). Sorteer op spelers nummer. Gebruik de exists operator.   | (<br>SELECT '1'<br>FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr<br>AND w.teamnr = t.teamnr))<br>ORDER BY 1   |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Maak een lijst met de spelers (naam van de speler, voorletter en woonplaats) die ooit gespeeld hebben voor een team dat nu in de tweede divisie speelt en waarvoor geen enkele boete betaald werd voor 1 januari 1981. Sorteer van voor naar achter, oplopend. Zorg dat er geen dubbels worden getoond.                                       | SELECT DISTINCT s.naam, s.voorletters, s.plaats<br>FROM spelers s INNER JOIN wedstrijden w USING<br>(spelersnr)<br>WHERE w.teamnr IN<br>(<br>SELECT teamnr<br>FROM teams<br>WHERE divisie = 'tweede')<br>AND s.spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM boetes<br>WHERE datum < '1-1-1981')<br>ORDER BY 1,2,3;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef de twee laagste bondnrs terug. (tip: dwz er zijn dus minder dan 2 bondsnr die kleiner zijn). Sorteer op bondnr. Zonder het gebruik van LIMIT.  | SELECT s.bondsnr<br>FROM spelers s<br>WHERE 2 ><br>(<br>SELECT count(*)<br>FROM spelers sub<br>WHERE sub.bondsnr < s.bondsnr)<br>AND s.bondsnr IS NOT NULL<br>ORDER BY 1;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef van elke boete het betalingsnr, het boetebedrag en het percentage dat het bedrag uitmaakt van de som van alle bedragen. Sorteer deze data op het betalingsnr. Zorg dat er maar twee getallen na de komma getoond worden (rond af). Sorteer van voor naar achter.   | SELECT b1.betalingsnr, b1.bedrag,<br>round (100*b1.bedrag/sum(b2.bedrag),2)<br>FROM boetes b1, boetes b2 GROUP BY b1.betalingsnr,<br>b1.bedrag ORDER BY 1,2,3;  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef alle spelers die geen enkele wedstrijd voor team 1 hebben gespeeld. Sorteer op naam, daarna op spelersnr.  | SELECT spelersnr, naam<br>FROM spelers<br>WHERE spelersnr NOT IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden<br>WHERE teamnr = 1)<br>ORDER BY 2,1;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Maak een lijst met de spelers (naam van de speler, voorletter en woonplaats) die ooit gespeeld hebben voor een team dat nu in de tweede divisie speelt en waarvoor geen enkele boete betaald werd voor 1 januari 1981. Geen dubbels, sorteer van voor naar achter.  | SELECT DISTINCT s.naam, s.voorletters, s.plaats<br>FROM spelers s NATURAL INNER JOIN wedstrijden w<br>WHERE w.teamnr IN<br>(SELECT t.teamnr<br>FROM teams t<br>WHERE divisie = 'tweede')<br>AND NOT EXISTS<br>(SELECT 2<br>FROM boetes b<br>WHERE b.datum < '1-1-1981'<br>AND b.spelersnr = s.spelersnr)<br>ORDER BY 1,2,3;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef een overzicht van alle spelers, gevolgd door alle bestuursleden, gesorteerd op jaar van toetreding of beginjaar van hun functie en vervolgens op spelersnr. Geen dubbels tonen.  | SELECT spelersnr AS veld1, naam AS veld2, jaartoe AS<br>veld3<br>FROM spelers<br>UNION<br>SELECT spelersnr AS veld1, functie AS veld2, EXTRACT<br>(YEAR FROM begin_datum) AS veld3<br>FROM bestuursleden<br>ORDER BY veld3, veld1   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef een lijst met alle spelersnrs, naam en het aantalwedstrijden ze gespeeld hebben en op een nieuwe lijn het aantal bestuursfuncties die ze hebben/hadden. Spelers die zowel wedstrijden gespeeld hebben als bestuurslid zijn, komen dus twee keer voor in het resultaat. Sorteer op spelersnr en aantal. Geen dubbels tonen.               | SELECT spelers.spelersnr AS nr, spelers.naam,<br>COUNT(*) AS aantal<br>FROM spelers<br>INNER JOIN wedstrijden USING(spelersnr)<br>GROUP BY spelers.spelersnr, spelers.naam<br>UNION<br>SELECT bestuursleden.spelersnr AS nr, spelers.naam,<br>COUNT(*) AS aantal<br>FROM bestuursleden<br>INNER JOIN spelers USING(spelersnr)<br>GROUP BY bestuursleden.spelersnr, spelers.naam<br>ORDER BY nr, aantal;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef een overzicht van de boetebedragen, aantal gewonnen en verloren sets en aantal verschillende functies. Bekijk de output voor de manier hoe het getoond moet worden. Sorteer van links naar rechts.- Gebruik int als datatype. Tip: Het lijkt onlogisch, maar zelfs NULL krijgt een datatype en kan niet impliciet wijzigen van datatype. | SELECT SUM(bedrag) AS boetebedrag, CAST(null AS int)<br>AS aantalgewonnen, CAST(null AS int) AS aantalverloren,<br>CAST(null AS int) AS aantalfuncties<br>FROM boetes<br>UNION<br>SELECT null AS boetebedrag, SUM(gewonnen) AS<br>aantalgewonnen, null AS aantalverloren, NULL AS<br>aantalfuncties<br>FROM wedstrijden<br>UNION<br>SELECT null AS boetebedrag, null AS aantalgewonnen,<br>SUM(verloren) AS aantalverloren, NULL AS aantalfuncties<br>FROM wedstrijden<br>UNION<br>SELECT null AS boetebedrag, null AS aantalgewonnen,<br>null AS aantalverloren, COUNT(DISTINCT functie) AS<br>aantalfuncties<br>FROM bestuursleden<br>ORDER BY 1,2,3,4; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef voor de spelers die bestuurslid en/of teamkapitein zijn hun naam en een oplistng van hun functienamen (huidig of verleden) en hun divisies waarvoor ze kapitein zijn. Sorteer op spelersnaam en naam.  | SELECT distinct spelers.naam as spelersnaam,<br>rommel.naam<br>FROM spelers<br>INNER JOIN (<br>SELECT functie AS naam, spelersnr<br>FROM bestuursleden<br>UNION   |  |

|      |                                     |  |  |  |
|------|-------------------------------------|--|--|--|
|      |                                     | Gebruik geen OUTER JOIN of WHERE.  | SELECT divisie AS naam, spelersnr<br>FROM teams<br>) AS rommel USING (spelersnr)<br>ORDER BY 1,2;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef een lijst van alle spelers die bestuurslid geweest zijn (of nu nog zijn) en/of een boete hebben gehad hun aantal boetes en hun laatst begonnen bestuursfunctie. Zorg dat spelers die boetes gehad hebben én bestuurder zijn (geweest) twee keer voorkomen in de kolom 'data' twee verschillende waarden.<br>Sorteer op naam, voorletters en data.<br>Gebruik de to_char functie voor het formaat van de geboortedatum (bv 12/12/1900).  | SELECT naam, voorletters, to_char(geb_datum, 'DD/MM/YYYY') AS geboortedatum, data.data<br>FROM spelers<br>INNER JOIN (<br>SELECT spelersnr, 'Boetes: '    CAST(COUNT(*) AS CHARACTER(20)) AS data<br>FROM boetes<br>GROUP BY spelersnr<br>UNION<br>SELECT spelersnr, functie AS data<br>FROM bestuursleden<br>WHERE begin_datum = (<br>SELECT MAX(begin_datum)<br>FROM bestuursleden alle<br>WHERE alle.spelersnr = bestuursleden.spelersnr<br>)<br>) AS data ON spelers.spelersnr = data.spelersnr<br>ORDER BY naam, voorletters, data;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | We zijn wat betreft boetes alleen geïnteresseerd als de speler geboren is voor 1963.<br><br>Verder uitleg:<br>Geef een lijst van ALLE spelers die bestuurslid geweest zijn en/of een boete hebben gehad, (alleen als ze geboren zijn voor 1963) en hun laatst begonnen bestuursfunctie.<br>Zorg ervoor dat spelers die een bestuursfunctie hebben/hadden, geboren zijn voor 1963, maar geen boete hebben gekregen, toch in het resultaat voorkomen met een extra lijn 'Geen boetes'.<br>Spelers die geen bestuurslid zijn geweest, maar een boete hebben gehad, komen één keer voor in het resultaat met hun aantal boetes.<br>Spelers die geen bestuurslid zijn geweest en geboren zijn na 1962, moeten ook één keer in het resultaat voorkomen met als data 'Gewone speler' (het kan zelfs zijn dat geen boete gekregen hebben, alle spelers moeten sowieso getoond worden).<br>Sorteer op naam, voorletters en data (let op de hoofdletters in het veld data).<br><br>Gebruik de to_char functie voor het formaat van de geboortedatum (bv 12/12/1900). | SELECT naam, voorletters, to_char(geb_datum, 'DD/MM/YYYY') AS geboortedatum,<br>CASE<br>WHEN data.data IS NULL THEN 'Gewone speler'<br>ELSE data.data<br>END AS data<br>FROM spelers<br>LEFT OUTER JOIN (<br>SELECT spelers.spelersnr,<br>CASE<br>WHEN COUNT(betalingsnr) = 0 THEN 'Geen boetes'<br>ELSE 'Boetes: '    CAST(COUNT(betalingsnr) AS CHARACTER(20))<br>END AS data<br>FROM spelers<br>LEFT OUTER JOIN boetes USING (spelersnr)<br>WHERE EXTRACT(YEAR FROM geb_datum) < 1963<br>GROUP BY spelersnr<br>UNION<br>SELECT spelersnr, functie AS data<br>FROM bestuursleden<br>WHERE begin_datum = (<br>SELECT MAX(begin_datum)<br>FROM bestuursleden alle<br>WHERE alle.spelersnr = bestuursleden.spelersnr<br>)<br>) AS data ON spelers.spelersnr = data.spelersnr<br>ORDER BY naam, voorletters, data; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef de naam en het spelersnummer van de spelers die ooit penningmeester geweest zijn van de club, die bovendien ooit een boete betaald hebben van meer dan 75 euro, en die ooit een wedstrijd gewonnen hebben met meer dan 2 sets verschil. Sorteer van voor naar achter.<br>Probeer gelijk of beter te doen dan "Unique (cost=100.38..100.54 rows=21 width=68)".   | SELECT DISTINCT s.naam, s.spelersnr<br>FROM ((spelers s INNER JOIN bestuursleden be USING (spelersnr)) INNER JOIN boetes bo USING (spelersnr))<br>INNER JOIN wedstrijden w USING (spelersnr)<br>WHERE bo.bedrag >= 75<br>AND be.functie = 'Penningmeester'<br>AND be.eind_datum IS NOT NULL<br>AND (w.gewonnen - w.verloren) > 1<br>ORDER BY 1,2   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding. Sorteer van voor naar achter.<br>Probeer gelijk of beter te doen dan "Sort (cost=33.16..33.66 rows=200 width=86)"  | SELECT s.spelersnr,s.naam,s.voorletters,s.jaartoe-<br>(SELECT avg(jaartoe)<br>FROM spelers) AS Verschil<br>FROM spelers s<br>ORDER BY 1,2,3,4  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Je kan per speler berekenen hoeveel boetes die speler heeft gehad en wat het totaalbedrag per speler is. Pas nu deze querye aan zodat per verschillend aantal boetes wordt getoond hoe vaak dit aantal boetes voorkwam. Sorteer van voor naar achter.<br>Probeer gelijk of beter te doen dan "Sort (cost=46.39..46.89 rows=200 width=8)".  | SELECT a, count(a)<br>FROM (SELECT count(bedrag) AS a<br>FROM boetes<br>GROUP BY spelersnr) b<br>GROUP BY a<br>ORDER BY 1,2  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef alle spelers die alfabetisch (dus naam en voorletters, in deze volgorde) voor speler 8 staan. Sorteer van voor naar achter.<br>Probeer zo goed of beter te doen dan "Sort (cost=24.31..24.47 rows=67 width=88)"   | SELECT spelersnr, naam, voorletters, geb_datum<br>FROM spelers<br>WHERE naam  voorletters <<br>(SELECT naam  voorletters<br>FROM spelers<br>WHERE spelersnr=8)<br>ORDER BY 1,2,3;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef van elke speler het spelersnr, de naam en het verschil tussen zijn of haar jaar van toetreding en het gemiddeld jaar van toetreding van de spelers die in dezelfde plaats wonen. Sorteer van voor naar achter. Toon 3 getallen na de komma, maximaal 2 voor de komma; gebruik een cast functie.   | SELECT s.spelersnr,s.naam,s.voorletters,cast (s.jaartoe-<br>avg as numeric(5,3))<br>FROM spelers s,<br>(SELECT plaats, avg(jaartoe) AS avg<br>FROM spelers<br>GROUP BY plaats) a<br>WHERE a.plaats=s.plaats<br>ORDER BY 1, 2, 3, 4   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Toon alle mogelijke combinaties van de letters 'x' en 'y'. Tip zie handboek, voorbeeld met cijfers. (Wat is een cartesisch product?). Sorteer  | SELECT A.letter    B.letter<br>FROM (SELECT 'x' AS letter UNION SELECT 'y') as A,<br>(SELECT 'x' AS letter UNION SELECT 'y') as B<br>ORDER BY 1  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag i | Geef van elke speler die minstens 1 wedstrijd gewonnen heeft voor team nr 1 en voor wie in totaal meer dan 100 euro aan boete betaald is, het spelersnummer, zijn naam, woonplaats en het totale boetebedrag. Sorteer van voor naar achter.  | SELECT s.spelersnr, s.naam, s.plaats, sum(b.bedrag)<br>FROM spelers s INNER JOIN boetes b on<br>(b.spelersnr=s.spelersnr)<br>WHERE s.spelersnr IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden w<br>WHERE gewonnen > verloren<br>AND teamnr=1)<br>GROUP BY s.spelersnr, s.naam, s.plaats<br>HAVING sum(b.bedrag)>100   |  |










|      |                                   |  |  |   |
|------|-----------------------------------|--|--|---|
|      |                                   |  | order by 1,2,3,4;  |   |
| 2023 | Databanken: DB2<br>Examenvraag i  | Geef alle spelers voor wie meer boetes zijn betaald dan dat ze wedstrijden hebben gespeeld. Sorteer van voor naar achter   | SELECT s.naam, s.voorletters, s.geb_datum<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>GROUP BY s.spelersnr, s.naam, s.voorletters, s.geb_datum<br>HAVING count(*) ><br>(<br>SELECT count(*)<br>FROM wedstrijden w<br>WHERE w.spelersnr = s.spelersnr)<br>order by 1,2,3   |    |
| 2023 | Databanken: DB2<br>Examenvraag i  | Geef de twee laagste bondsnr's terug. (tip: dwz er zijn dus minder dan 2 bondsnr die kleiner zijn) Sorteer van voor naar achter.   | SELECT s.bondsnr<br>FROM spelers s<br>WHERE 2 ><br>(<br>SELECT count(*)<br>FROM spelers sub<br>WHERE sub.bondsnr < s.bondsnr)<br>AND s.bondsnr IS NOT NULL<br>order by 1   |    |
| 2023 | Databanken: DB2<br>Examenvraag i  | Geef een lijst van alle huidige bestuursleden die nog geen boete gekregen hebben, maar wel al minstens één wedstrijd verloren hebben. Sorteer op spelersnr en wedstrijdnr.   | SELECT S.spelersnr, S.naam, B.functie, B.begin_datum, W.wedstrijdnr<br>FROM spelers S<br>INNER JOIN bestuursleden B ON S.spelersnr = B.spelersnr AND B.eind_datum IS NULL<br>INNER JOIN wedstrijden W ON S.spelersnr = W.spelersnr AND W.gewonnen < W.verloren<br>LEFT OUTER JOIN boetes BO ON S.spelersnr = BO.spelersnr<br>WHERE BO.betalingsnr IS NULL<br>ORDER BY S.spelersnr, W.wedstrijdnr   |    |
| 2023 | Databanken: DB2<br>Examenvraag i  | Selecteer de namen van voorzitters en de periode (= begindatum streepje einddatum) waarin ze voorzitter waren (of nog steeds zijn) en op welke datum ze beboet zijn geweest. Gebruik   .   | select s.naam, CASE WHEN be.eind_datum IS NOT NULL THEN to_char(be.begin_datum, 'DD/MM/YYYY')    ' - '    to_char(be.eind_datum, 'DD/MM/YYYY') ELSE to_char(be.begin_datum, 'DD/MM/YYYY')    ' - ' END as periode, bo.datum<br>from bestuursleden as be, boetes as bo, spelers as s<br>where be.spelersnr = bo.spelersnr<br>and functie = 'Voorzitter'<br>and s.spelersnr = be.spelersnr   |    |
| 2023 | Databanken: DB2<br>Examenvraag i  | Geef een lijst van alle mogelijke dubbelcombinaties van spelers. Beperking: de verschillende spelers mogen geen kapitein zijn van hetzelfde team (moest dit mogelijk zijn). Orden eerst naar speler1, dan naar speler2.  | select DISTINCT s.voorletters    ' '    s.naam AS "speler1", t.voorletters    ' '    t.naam AS "speler2"<br>from spelers as s, spelers as t, teams as te, teams as tt<br>where s.spelersnr <> t.spelersnr AND NOT (s.spelersnr = te.spelersnr AND t.spelersnr = tt.spelersnr AND te.teamnr = tt.teamnr)<br>ORDER BY 1,2  |   |
| 2023 | Databanken: DB2<br>Examenvraag i  | Geef van elke speler die enkel wedstrijden gewonnen heeft voor team nr 1 en voor wie in totaal meer dan 100 euro aan boete betaald is, het spelersnummer, zijn naam, woonplaats en het totale boetebedrag. Dit resultaat moet aflopend geordend worden op het totale boetebedrag. Sorteer van voor naar achter.  | SELECT s.spelersnr, s.naam, s.plaats, sum(b.bedrag)<br>FROM spelers s INNER JOIN boetes b USING (spelersnr)<br>WHERE s.spelersnr IN<br>(<br>SELECT spelersnr<br>FROM wedstrijden w<br>WHERE gewonnen > verloren<br>AND teamnr=1)<br>GROUP BY s.spelersnr, s.naam, s.plaats<br>HAVING sum(b.bedrag)>100<br>order by 1,2,3,4   |  |
| 2023 | Databanken: DB2<br>Examenvraag ii | Geef voor elke een klant een overzicht aan uitgaven. Hoe? Geef voor elke klant een cumulatie overzicht van prijs van de reizen waar hij aan deelgenomen heeft. De volgorde, waarin er wordt cumulatief wordt opgeteld, wordt bepaald door de vertrekdatum van de reis. Sorteer van voor naar achter. Tip: vergelijk deze uitvoer met de uitvoer van een query die een gesorteerd overzicht geeft van de klanten en hun deelnames aan reizen. | select klantnr, reisnr, sum(prijs) OVER (partition by klantnr order by vertrekdatum)<br>from reizen natural join deelnames<br>order by 1, 2, 3;  |  |
| 2023 | Databanken: DB2<br>Examenvraag ii | Geef het aantal unieke satellieten van planeten binnen ons sterrenstelsel  | SELECT count(DISTINCT S.objectnaam) as aantal_satellieten<br>FROM hemelobjecten Z, hemelobjecten P, hemelobjecten S<br>WHERE Z.satellietvan is null AND P.satellietvan = Z.objectnaam AND S.satellietvan = P.objectnaam  |  |
| 2023 | Databanken: DB2<br>Examenvraag ii | Geef de diameter van de grootste, niet bezochte maan (satelliet van een planeet).  | SELECT MAX(manen.diameter) AS grootstemaan<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten manen ON planeten.objectnaam = manen.satellietvan AND planeten.satellietvan = 'Zon'<br>LEFT OUTER JOIN bezoeken ON manen.objectnaam = bezoeken.objectnaam<br>WHERE bezoeken.reisnr IS NULL;  |  |
| 2023 | Databanken: DB2<br>Examenvraag ii | Vind per planeet die zijn maan die het verst ervan verwijderd is. Geef planeet, maan en afstand en sorteer stijgend op afstand.  | SELECT planeet.objectnaam as planeet, M.objectnaam as maan, M.afstand as afstand<br>FROM hemelobjecten ster<br>INNER JOIN hemelobjecten planeet ON ster.satellietvan IS NULL AND planeet.satellietvan = ster.objectnaam<br>INNER JOIN hemelobjecten maan ON maan.satellietvan = planeet.objectnaam<br>INNER JOIN hemelobjecten M on M.satellietvan = planeet.objectnaam<br>GROUP BY planeet.objectnaam, M.objectnaam<br>HAVING M.afstand = MAX(maan.afstand)<br>ORDER BY afstand |  |
|      |                                   | Hoeveel kilometers heeft iedereen in totaal gevlogen tot nu toe en   | SELECT k.klantnr, k.naam    ' '    k.vnaam AS naam, tot_bedrag, tot_afstand,<br>CASE<br>WHEN tot_afstand <> 0 THEN<br>CAST(tot_bedrag/tot_afstand AS varchar)<br>ELSE 'veel geld voor niks of niet op reis geweest'  |   |


|      |                                      |   |   |   |
|------|--------------------------------------|---|---|---|
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | <p>hoeveel hebben ze hier in totaal voor betaald. Vermits we de posities van de planeten niet kennen, mag je de afstanden van de hemelobjecten direct gebruiken. Geef het totaal gespendeerde bedrag, de afgelegde kilometers, de prijs per kilometer en datum van hun laatste vlucht van al hun persoonlijke reizen. In het geval dat iemand niet op reis is geweest of geen kilometers gedaan heeft, toon je de boodschap 'veel geld voor niks of niet op reis geweest' in de kolom prijs_per_kilometer.</p> <p>Sorteer van voor naar achter.</p> | <pre>END AS prijs_per_kilometer, laatste_reis_datum FROM klanten k NATURAL LEFT OUTER JOIN ((SELECT klantnr, sum(prijs) AS tot_bedrag, max(vertrekdatum) AS laatste_reis_datum FROM deelnames NATURAL LEFT OUTER JOIN reizen GROUP BY klantnr) AS dr NATURAL LEFT OUTER JOIN (SELECT klantnr, sum(afstand) AS tot_afstand FROM (deelnames NATURAL INNER JOIN bezoeken) NATURAL INNER JOIN hemelobjecten GROUP BY klantnr) AS bh) ORDER BY 1,2,3,4,5,6;</pre>  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | <p>Geef de deelnemers waarbij hun aantal reizen die ze ondernemen groter is dan alle hemelobjecten (die niet beginnen met de letter 'M') hun aantal keren dat ze bezocht zijn. Of anders geformuleerd: Geef de deelnemers met meer deelnames dan het grootste aantal bezoeken aan een hemelobject dat niet met de letter 'M' begint (:deze deelnemer meer deelnames heeft dan de "grootste" .. = deze deelnemer heeft meer deelnames dan "alle" ..)</p> <p>Sorteer op klantnr.</p>  | <pre>SELECT klantnr, vnaam, naam, COUNT(*) AS aantaldeelnames FROM klanten INNER JOIN deelnames using (klantnr) GROUP BY klantnr, vnaam, naam HAVING COUNT(*) &gt; ALL ( SELECT COUNT(*) FROM bezoeken WHERE objectnaam NOT LIKE 'M%' GROUP BY objectnaam ) ORDER BY klantnr,2,3,4;</pre>   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | <p>Geef voor elke geboorteejaar van de klanten, het aantal klanten, het kleinste klantnummer en het grootste klantnummer. Geef ook het totaal aantal klanten en het kleinste en grootste klantnummer.</p> <p>Sorteer van voor naar achter.</p>  | <pre>SELECT extract(year from geboortedatum), count(*),min(klantnr),max(klantnr) FROM klanten GROUP BY ROLLUP (extract(year from geboortedatum)) ORDER BY 1,2,3,4;</pre>  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | <p>Geef het reisnr, de prijs en vertrekdatum van de reis met de hoogste gemiddelde verblijfsduur op een hemelobject (=som van de verblijfsduur / aantal bezoeken per reis).</p>   | <pre>SELECT reizen.reisnr, prijs, vertrekdatum FROM reizen INNER JOIN bezoeken USING (reisnr) GROUP BY reizen.reisnr, prijs, vertrekdatum HAVING (SUM(verblijfsduur) / COUNT(objectnaam)) = ( SELECT MAX(gemiddelde) FROM ( SELECT SUM(verblijfsduur) / COUNT(objectnaam) as gemiddelde FROM bezoeken GROUP BY reisnr ) AS reisgemiddelden )</pre>  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | <p>Geef de naam, diameter en het langste verblijf op van alle hemelobjecten met minder dan 5 hemelobjecten die rond dit object draaien. (ze moeten dus bezocht zijn)</p> <p>Sorteer op objectnaam.</p>  | <pre>SELECT hemelobjecten.objectnaam, hemelobjecten.diameter, MAX(bezoeken.verblijfsduur) AS maximale_verblijf FROM bezoeken INNER JOIN hemelobjecten ON bezoeken.objectnaam = hemelobjecten.objectnaam LEFT OUTER JOIN hemelobjecten manen ON hemelobjecten.objectnaam = manen.satellietvan GROUP BY hemelobjecten.objectnaam, hemelobjecten.diameter HAVING COUNT(DISTINCT manen.objectnaam) &lt; 5 ORDER BY objectnaam;</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | <p>Maak een lijst van de mensen die Mars wel bezocht hebben maar lo nog niet. Sorteer van voor naar achter.</p>   | <pre>SELECT k.klantnr, k.naam FROM ((klanten k INNER JOIN deelnames d using (klantnr)) INNER JOIN reizen r using(reisnr)) INNER JOIN bezoeken b using(reisnr) WHERE b.objectnaam = 'Mars' EXCEPT SELECT k.klantnr, k.naam FROM ((klanten k INNER JOIN deelnames d using (klantnr)) INNER JOIN reizen r using(reisnr)) INNER JOIN bezoeken b using(reisnr) WHERE b.objectnaam = 'Io' order by 1,2</pre>  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | <p>Geef de planeet (draait dus rond de zon) met de meeste satellieten.</p> <p>Sorteer op objectnaam.</p>  | <pre>SELECT planeten.objectnaam FROM hemelobjecten planeten LEFT OUTER JOIN hemelobjecten satellieten ON planeten.objectnaam = satellieten.satellietvan WHERE planeten.satellietvan = 'Zon' GROUP BY planeten.objectnaam HAVING COUNT(*) = ( SELECT MAX(aantalsatellieten) FROM ( SELECT COUNT(*) AS aantalsatellieten FROM hemelobjecten planeten LEFT OUTER JOIN hemelobjecten satellieten ON planeten.objectnaam = satellieten.satellietvan WHERE planeten.satellietvan = 'Zon' GROUP BY planeten.objectnaam ) AS aantallen ) ORDER BY planeten.objectnaam</pre> |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | <p>Geef voor elk hemelobject de minimale en maximale gemiddelde afstand tot zijn zon (de centrale ster in een sterrenstelsel) als je weet dat de kolom 'afstand' de gemiddelde afstand bevat tot het hemelobject waarond ze draaien. Met de grootte van het hemelobject hoeft je geen rekening te houden.</p> <p>Sorteer op minimale afstand en op objectnaam.</p> <p>Tip: bekijk de functie COALESCE om met null-waardes rekening te houden. Bv coalesce(null - 7,0) = 0 , maar coalesce(null,0)-coalesce(7,0) is -7</p>                           | <pre>SELECT hemelobjecten.objectnaam, coalesce(hemelobjecten.afstand, 0) + coalesce(draaitrond.afstand, 0) AS maximale_afstand, abs(coalesce(draaitrond.afstand, 0) - coalesce(hemelobjecten.afstand, 0)) AS minimale_afstand FROM hemelobjecten LEFT OUTER JOIN hemelobjecten draaitrond ON draaitrond.objectnaam = hemelobjecten.satellietvan ORDER BY minimale_afstand, objectnaam;</pre>  |  |
|      |                                      |   | <pre>SELECT MAX(diameter) AS grootste FROM hemelobjecten INNER JOIN bezoeken USING (objectnaam)</pre>   |   |


|      |                                      |  |   |   |
|------|--------------------------------------|--|---|---|
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef de diameter van het grootste hemellichaam dat bezocht is op de vroegste reis waar klantnr 126 niet op meegegaan is.   | INNER JOIN reizen USING (reisnr)<br>WHERE vertrekdatum = (<br>SELECT MIN(vertrekdatum)<br>FROM reizen<br>INNER JOIN bezoeken USING (reisnr)<br>WHERE<br>reisnr NOT IN (<br>SELECT reisnr<br>FROM deelnames<br>WHERE klantnr = 126<br>)<br>)   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef de hemelobjecten die nog nooit bezocht zijn (een bezoek heeft een verblijfsduur van minstens één dag).  | SELECT hemelobjecten.objectnaam<br>FROM hemelobjecten<br>LEFT OUTER JOIN bezoeken ON<br>hemelobjecten.objectnaam = bezoeken.objectnaam AND<br>bezoeken.verblijfsduur > 0<br>WHERE bezoeken.objectnaam IS NULL<br>ORDER BY hemelobjecten.objectnaam;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | We willen telkens het aantal reizen en de totale prijs van de volgende situaties. Voor elke maand van vertrek, voor elk tiental van de reisduur, voor de combinatie van maand van vertrek en tiental van de reisduur en het het totale plaatje. Sorteer van voor naar achter.  | SELECT extract(month from<br>vertrekdatum),round(reisduur/10), count(*), sum(prijs)<br>FROM reizen<br>GROUP BY CUBE (extract(month from<br>vertrekdatum),round(reisduur/10))<br>ORDER by 1,2,3,4;   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Sorteer de klanten aflopend op gemiddelde kostprijs per bezoek (totaalprijs van alle reizen per klant/totaal aantal dagen verblijf op een hemelobject), op twee cijfers na de komma afgerond, daarna op klantnr.   | SELECT klanten.klantnr,<br>ROUND(SUM(reizen.prijs)/SUM(bezoeken.verblijfsduur),2)<br>AS gemiddeldeprijs<br>FROM klanten<br>INNER JOIN deelnames ON klanten.klantnr =<br>deelnames.klantnr<br>INNER JOIN reizen ON deelnames.reisnr = reizen.reisnr<br>INNER JOIN bezoeken ON reizen.reisnr =<br>bezoeken.reisnr<br>GROUP BY klanten.klantnr<br>ORDER BY gemiddeldeprijs DESC, klantnr;              |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef het hemellichaam dat het laatst bezocht is. Gebruik hiervoor de laatste vertrekdatum van de reis en laatste volgnummer van bezoek. Tip: gebruik hiervoor een rij-subquery. Gebruik geen limit of top.   | SELECT objectnaam<br>FROM bezoeken<br>WHERE (reisnr, volgnr) = (<br>SELECT bezoeken.reisnr, MAX(bezoeken.volgnr)<br>FROM bezoeken<br>INNER JOIN reizen laatstereis USING (reisnr)<br>WHERE laatstereis.vertrekdatum = (<br>SELECT MAX(vertrekdatum)<br>FROM reizen<br>)<br>)<br>GROUP BY bezoeken.reisnr<br>)   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef voor de hemelobjecten een diameter groter dan 1000 het aantal hemelobjecten en de gemiddelde diameter per groep die aan het volgende voldoet. We zien per object zien hoeveel satellieten hieraan voldoen, daarnaast in combinatie met hetzelfde object en afstand per 100tal. Alsook een algemeen overzicht. Sorteer van voor naar achter. | SELECT objectnaam, round(afstand/100),satellietvan,<br>count(*), avg(diameter)<br>FROM hemelobjecten<br>WHERE diameter > 1000<br>GROUP BY ROLLUP (satellietvan,<br>(objectnaam,round(afstand/100)));  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Hoe lang was het geleden dat er nog een reis vertrokken was? Geef daarnaast de totale reisduur per jaar incrementeel in de tijd (hier genaamd jaar_duur). Sorteer op reisnr en de andere kolommen.   | SELECT reisnr, lag(reisnr) OVER w1 as vorig_reisnr,<br>vertrekdatum,<br>vertrekdatum - lag(vertrekdatum) OVER w1 as tussen_tijd,<br>reisduur,<br>extract(year from vertrekdatum) as jaar,<br>sum(reisduur)<br>OVER (partition by extract(year from vertrekdatum) order<br>by vertrekdatum)<br>as jaar_duur<br>FROM reizen<br>WINDOW w1 as (order by vertrekdatum)<br>ORDER BY 1,2,3,4,5,6,7;        |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Maak een lijst van klanten die meer dan 2 keer een reis gemaakt hebben waarbij er geen bezoek was aan Jupiter.   | SELECT k.klantnr, (k. naam    ' '    k.vnaam) AS klantnaam,<br>COUNT(d.reisnr) AS aantalreizen<br>FROM klanten k<br>INNER JOIN deelnames d USING (k.klantnr)<br>WHERE d.reisnr NOT IN<br>(SELECT b.reisnr<br>FROM bezoeken b INNER JOIN hemelobjecten o ON<br>b.objectnaam = o.objectnaam<br>WHERE o.objectnaam = 'Jupiter')<br>GROUP BY k.klantnr, k. naam, k.vnaam<br>HAVING count(d.reisnr) > 2; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef een lijst van alle hemelobjecten die meer keer bezocht gaan worden dan Jupiter. (onafhankelijk van het aantal deelnames) Sorteer op objectnaam.   | SELECT hemelobjecten.objectnaam,<br>hemelobjecten.diameter<br>FROM hemelobjecten<br>INNER JOIN bezoeken USING (objectnaam)<br>GROUP BY hemelobjecten.objectnaam,<br>hemelobjecten.diameter<br>HAVING COUNT(*) > (<br>SELECT COUNT(*)<br>FROM bezoeken<br>WHERE objectnaam = 'Jupiter'<br>)<br>ORDER BY hemelobjecten.objectnaam   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef de klantnr voor de klant met het meeste bezoeken aan de maan. Geef ook het aantal bezoeken. Gebruik geen limit of top.  | SELECT d.klantnr, count(b.objectnaam)<br>FROM deelnames d<br>INNER JOIN reizen r ON d.reisnr = r.reisnr<br>INNER JOIN bezoeken b ON r.reisnr = b.reisnr<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr<br>HAVING COUNT(b.objectnaam) = (<br>SELECT MAX(c)<br>FROM (<br>SELECT d.klantnr, COUNT(b.objectnaam) AS c<br>FROM deelnames d   |  |




|      |                                      |   |  |   |
|------|--------------------------------------|---|--|---|
|      |                                      |   | INNER JOIN bezoeken b ON d.reisnr = b.reisnr<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr<br>) AS temp<br>);   |   |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef een lijst van alle planeten in het heelal die minstens 1 satelliet hebben met een diameter groter dan 2000km (je weet dus niet hoeveel satellieten, kan ook bv 100 zijn, het aantal satelieten bestaat maximaal uit 3 cijfers, gebruik dit bij je omzetting). Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer op planeetnaam.<br>Gebruik to_char .. 999 om te casten voor de count.                              | SELECT planeet.objectnaam, CASE WHEN<br>count(sat.objectnaam) = 0 THEN 'Geen' ELSE<br>to_char(count(sat.objectnaam), '999') END AS<br>"interessante satellieten"<br>FROM hemelobjecten planeet INNER JOIN hemelobjecten<br>ster ON (planeet.satellietvan = ster.objectnaam AND<br>ster.satellietvan is null)<br>LEFT OUTER JOIN hemelobjecten sat ON<br>(sat.satellietvan = planeet.objectnaam AND sat.diameter ><br>2000)<br>GROUP BY planeet.objectnaam<br>ORDER BY planeet.objectnaam                                       |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef voor elke reis het aantal klanten waarvan de naam niet met een 'G' begint en waarvan de periode van de geboortedatum van de klant tot de vertrekdatum van de reis overlapt met de huidige datum en 50 jaar verder (gebruik hiervoor de gepaste operator: OVERLAPS).<br>Indien er op de reis hemelobjecten worden bezocht waarvan de tweede letter van het hemelobject voorkomt in de naam van het hemelobject waarvan dit bezocht hemelobject een satelliet is, dan wordt deze reis genegeerd.<br>Sorteer op reisnr. | SELECT r.reisnr, count(k.klantnr)<br>FROM (REIZEN r NATURAL LEFT OUTER JOIN<br>deelnames)<br>LEFT OUTER JOIN klanten k ON (k.klantnr =<br>deelnames.klantnr)<br>AND upper(naam) NOT LIKE 'G%'<br>AND (geboortedatum,vertrekdatum) OVERLAPS<br>(CURRENT_DATE,CURRENT_DATE+interval '50 year')<br>WHERE r.reisnr NOT IN<br>(SELECT reisnr<br>FROM bezoeken NATURAL INNER JOIN hemelobjecten<br>WHERE lower(satellietvan) LIKE '% '   <br>lower(substring(objectnaam FROM 2 FOR 1))    '%')<br>GROUP BY r.reisnr<br>ORDER BY 1,2; |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef een lijst van alle planeten die minstens 1 satelliet hebben op een afstand groter dan 500km. Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Bij hen zet je als output: "Geen". Sorteer omgekeerd alfabetisch op planeetnaam.<br><br>Gebruik to_char van vaste lengte met 3 tekens voor de tweede kolom als het een cijfer is.  | SELECT planeet.objectnaam, CASE WHEN<br>count(sat.objectnaam) = 0 THEN 'Geen' ELSE<br>to_char(count(sat.objectnaam), '999') END AS "ver<br>verwijderde satellieten"<br>FROM hemelobjecten planeet INNER JOIN hemelobjecten<br>ster ON (planeet.satellietvan = ster.objectnaam AND<br>ster.satellietvan is null)<br>LEFT OUTER JOIN hemelobjecten sat ON<br>(sat.satellietvan = planeet.objectnaam AND sat.afstand ><br>500)<br>GROUP BY planeet.objectnaam<br>ORDER BY planeet.objectnaam desc                                 |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef per reis incrementeel de totale verblijfsduur volgens de volgorde waarin de hemelobjecten bezocht worden. Geef ook de totale verblijfsduur van alle reizen om alles in perspectief te zetten. Sorteer op reisnr, volgnr, objectnaam, verblijfsduur, de volgende kolommen.  | SELECT reisnr, volgnr, objectnaam, verblijfsduur,<br>sum(verblijfsduur)<br>OVER(partition by reisnr order by volgnr) as inc_duur,<br>sum(verblijfsduur) over () as tot_duur<br>FROM bezoeken<br>ORDER by 1,2,3,4,5,6;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef een lijst van alle planeten die minstens 1 satelliet hebben wiens naam een kleine letter i bevat. Voor deze planeten toon je het aantal dergelijke satellieten. Geef ook een lijst van de andere planeten. Gebruik to_char van 3 cijfers voor de omzetting. Bij hen zet je als output: "Geen". Sorteer op planeetnaam.   | SELECT planeet.objectnaam, CASE WHEN<br>count(sat.objectnaam) = 0 THEN 'Geen' ELSE<br>to_char(count(sat.objectnaam), '999') END AS "satellieten<br>met een letter i"<br>FROM hemelobjecten planeet INNER JOIN hemelobjecten<br>ster ON (planeet.satellietvan = ster.objectnaam AND<br>ster.satellietvan is null)<br>LEFT OUTER JOIN hemelobjecten sat ON<br>(sat.satellietvan = planeet.objectnaam AND<br>sat.objectnaam LIKE '%i%')<br>GROUP BY planeet.objectnaam<br>ORDER BY planeet.objectnaam                             |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef de naam, diameter en het langste verblijf op alle hemelobjecten met minder dan 5 hemelobjecten die rond dit object draaien. Geef enkel hemelobjecten die bezocht zijn of waar gepasseerd wordt.<br>Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam,<br>hemelobjecten.diameter, MAX(bezoeken.verblijfsduur) AS<br>maximale_verblijf<br>FROM bezoeken<br>INNER JOIN hemelobjecten ON bezoeken.objectnaam =<br>hemelobjecten.objectnaam<br>LEFT OUTER JOIN hemelobjecten manen ON<br>hemelobjecten.objectnaam = manen.satellietvan<br>GROUP BY hemelobjecten.objectnaam,<br>hemelobjecten.diameter<br>HAVING COUNT(DISTINCT manen.objectnaam) < 5<br>ORDER BY objectnaam;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef de volledige frequentietabel voor de diameters van de hemelobjecten (frequentie: hoeveel oecten zijn er met de gegeven diameter, cumulatieve Frequentie, relatieve frequentie, Relatieve cumulatieve frequentie). Let op de datatypes en de precisie, gebruik CAST, rond niet af. Sorteer op diameter.   | SELECT b.diameter AS diameter, f, count(*) as "cf",<br>CAST(f*100.00/tot AS NUMERIC(5,2))AS rf,<br>CAST(100.00*count(*)/tot AS NUMERIC(5,2)) as "crf"<br>FROM (SELECT diameter, count(*) AS f<br>FROM hemelobjecten<br>GROUP BY diameter) AS b<br>INNER JOIN hemelobjecten bo ON (b.diameter >=<br>bo.diameter),<br>(SELECT count(*) as tot FROM hemelobjecten) AS boe<br>GROUP BY b.diameter, f, tot<br>ORDER BY 1;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef alle niet-bezochte hemelobjecten, buiten het grootste hemellichaam.<br>Sorteer op diameter en objectnaam.  | SELECT objectnaam, afstand, diameter<br>FROM hemelobjecten<br><br>WHERE diameter < ANY (<br>SELECT diameter<br>FROM hemelobjecten rest<br>WHERE rest.objectnaam <> hemelobjecten.objectnaam<br>)<br><br>AND NOT EXISTS (<br>SELECT *<br>FROM bezoeken<br>WHERE bezoeken.objectnaam =<br>hemelobjecten.objectnaam<br>)<br>ORDER BY diameter, objectnaam;  |  |

|      |                                      |   |  |   |
|------|--------------------------------------|---|--|---|
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef de planeten en het aantal verschillende personen die hen bezocht hebben (met of zonder verblijf).<br>Sorteer op objectnaam.  | SELECT hemelobjecten.objectnaam, COUNT(DISTINCT deelnames.klantnr) AS aantalbezoekers<br>FROM hemelobjecten<br>LEFT OUTER JOIN bezoeken ON<br>hemelobjecten.objectnaam = bezoeken.objectnaam<br>LEFT OUTER JOIN deelnames ON bezoeken.reisnr =<br>deelnames.reisnr<br>WHERE hemelobjecten.satellietvan = 'Zon'<br>GROUP BY hemelobjecten.objectnaam<br>ORDER BY hemelobjecten.objectnaam;  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef de klant die het meest op de maan is geweest (+het aantal).<br>Sorteer van voor naar achter.   | SELECT d.klantnr, count(b.objectnaam)<br>FROM (deelnames d INNER JOIN reizen r USING (reisnr))<br>INNER JOIN bezoeken b USING (reisnr)<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr<br>HAVING count(b.objectnaam) =<br>(SELECT max(c)<br>FROM (SELECT d.klantnr, count(b.objectnaam) AS c<br>FROM (deelnames d INNER JOIN reizen r USING (reisnr))<br>INNER JOIN bezoeken b<br>USING (reisnr)<br>WHERE b.objectnaam = 'Maan'<br>GROUP BY d.klantnr) AS temp)<br>order by 1,2   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Maak een lijst met die mensen die meer dan 2 maal een reis ondernomen hebben waarin men geen enkele satelliet van Jupiter bezoekt !. Sorteer van voor naar achter.  | SELECT k.klantnr, (k.naam    k.vnaam) AS "Volledige naam", count(d.reisnr) AS "Aantal Ondernomen Reizen"<br>FROM klanten k JOIN deelnames d USING (k.klantnr)<br>WHERE d.reisnr NOT IN<br>(SELECT b.reisnr<br>FROM bezoeken b NATURAL INNER JOIN hemelobjecten o<br>WHERE o.satellietvan = 'Jupiter')<br>GROUP BY k.klantnr, k.naam    k.vnaam<br>HAVING count(d.reisnr)>2<br>order by 1,2,3   |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef de gemiddelde afstanden tot hun planeet van alle satellieten van Saturnus, afgerond op 2 getallen na de komma  | SELECT round(AVG(S.afstand), 2) as "Gemiddelde afstand"<br>FROM hemelobjecten P, hemelobjecten S<br>WHERE P.objectnaam = 'Saturnus' AND S.satellietvan = P.objectnaam  |    |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Welke planeten met meer dan 7 manen worden bezocht (met of zonder verblijf)?<br>Sorteer aflopend op basis van het aantal manen.<br>Let erop dat je planeten die meerdere keren bezocht worden, niet dubbel telt.  | SELECT planeten.objectnaam<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten satellieten ON<br>satellieten.satellietvan = planeten.objectnaam<br>INNER JOIN hemelobjecten sterren ON<br>planeten.satellietvan = sterren.objectnaam<br>INNER JOIN bezoeken ON planeten.objectnaam =<br>bezoeken.objectnaam<br>WHERE sterren.satellietvan IS NULL<br>GROUP BY planeten.objectnaam<br>HAVING COUNT(DISTINCT satellieten.objectnaam) > 7<br>ORDER BY COUNT(DISTINCT satellieten.objectnaam)<br>DESC;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef voor elk hemelobject de minimale en maximale gemiddelde afstand tot zijn zon (de centrale ster in een sterrenstelsel) als je weet dat de kolom 'afstand' de gemiddelde afstand bevat tot het hemelobject waarond ze draaien. Met de grootte van het hemelobject hoeft je geen rekening te houden.<br>Sorteer op minimale afstand en op objectnaam. | SELECT hemelobjecten.objectnaam,<br>CASE<br>WHEN hemelobjecten.afstand IS NULL THEN 0<br>WHEN draaitrond.afstand IS NULL THEN<br>hemelobjecten.afstand<br>ELSE draaitrond.afstand + hemelobjecten.afstand<br>END AS maximale_afstand,<br>CASE<br>WHEN hemelobjecten.afstand IS NULL THEN 0<br>WHEN draaitrond.afstand IS NULL THEN<br>hemelobjecten.afstand<br>ELSE draaitrond.afstand - hemelobjecten.afstand<br>END AS minimale_afstand<br>FROM hemelobjecten<br>LEFT OUTER JOIN hemelobjecten draaitrond ON<br>hemelobjecten.satellietvan = draaitrond.objectnaam<br>ORDER BY minimale_afstand, objectnaam; |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Welke planeten met meer dan 7 manen worden bezocht (met of zonder verblijf)?<br>Sorteer aflopend op basis van het aantal manen.<br>Let erop dat je planeten die meerdere keren bezocht worden, niet dubbel telt.  | SELECT planeten.objectnaam<br>FROM hemelobjecten planeten<br>INNER JOIN hemelobjecten satellieten ON<br>satellieten.satellietvan = planeten.objectnaam<br>INNER JOIN bezoeken ON planeten.objectnaam =<br>bezoeken.objectnaam<br>WHERE planeten.satellietvan = 'Zon'<br>GROUP BY planeten.objectnaam<br>HAVING COUNT(DISTINCT satellieten.objectnaam) > 7<br>ORDER BY COUNT(DISTINCT satellieten.objectnaam)<br>DESC;  |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef per klant het totaal aantal reizen waaraan deze klant zal deelnemen en het langste bezoek dat deze klant zal maken aan een hemelobject, over alle reizen heen.<br>Klanten zonder reizen of zonder bezoeken moeten ook voorkomen in het overzicht.<br>Sorteer op klantnr.   | SELECT klanten.klantnr, COUNT(DISTINCT reizen.reisnr)<br>AS aantal, MAX(bezoeken.verblijfsduur) AS langstebezoek<br>FROM klanten<br>LEFT OUTER JOIN deelnames ON klanten.klantnr =<br>deelnames.klantnr<br>LEFT OUTER JOIN reizen ON deelnames.reisnr =<br>reizen.reisnr<br>LEFT OUTER JOIN bezoeken ON deelnames.reisnr =<br>bezoeken.reisnr<br>GROUP BY klanten.klantnr<br>ORDER BY klanten.klantnr;   |  |
| 2023 | Databanken:<br>DB2<br>Examenvraag ii | Geef enkel de 2 voorlaatste reizen terug. De positie van reizen wordt bepaalt door het reisnummer.<br>Ter vergelijking, als je de getallen 1 t/m 10 neemt, dan is dit 8 en 9. Sorteer van voor naar achter.<br>Gebruik enkel ISO sql.   | SELECT *<br>from<br>(select * from reizen order by reisnr desc offset 1 fetch first 2 rows only) as t<br>order by 1,2,3,4;   |  |
|      |                                      | Welke van de volgende stellingen zijn <b>JUIST</b> ?:   |  |   |

|      |                                       |  |   |
|------|---------------------------------------|--|---|
| 2023 | Databanken:<br>DB2<br>Examenvraag iii | <p>A Relationale databanken zijn achterhaald.<br/>B PG_BENCH kan je gebruiken om de kostprijs van queries te berekenen<br/>C Een MongoDB installatie voldoet aan tegelijk aan C A en P<br/>D Een goeie PostgreSQL installatie voldoet tegelijk aan C A en P<br/>E Een standaard MySQL installatie voldoet niet tegelijk aan C A en P<br/>F PostgreSQL is een zuiver relationele databank<br/>G PGAdmin is een beperkte relationele databank<br/>H NoSQL databanken zijn altijd sneller dan relationele databanken<br/>I NoSQL voldoet aan meestal aan B, A, S en E<br/>J Replicatie vervangt backup, backup is dus niet nodig als er replicatie is.<br/>K Een expliciete INNER JOIN geeft betere prestatie dan dezelfde conditie in de WHERE</p> <p>Dit is geen sql vraag, maar een meerkeuze vraag. Je antwoord met de letters die van toepassing zijn. (2p)</p> <p>bv</p> <p>BDE</p> <p>Opm:<br/>* Geldige oplossingen bestaan uit hoofdletters zonder gebruik van spaties of andere leestekens, alfabetisch gesorteerd.<br/>* Voorbeelden van ONGELDIGE oplossingen: BA, b a, 'AB', AB(spatie) .</p> <p>PS: Het is normaal dat sqldropbox een fout geeft, je kan dit negeren.</p> |  |
|------|---------------------------------------|--|---|

|      |                                       |   |  |
|------|---------------------------------------|---|--|
| 2023 | Databanken:<br>DB2<br>Examenvraag iii | <p>Welke van de volgende stellingen zijn JUIST?:</p> <p>A Relationale databanken zijn achterhaald.<br/>B PG_BENCH kan je gebruiken om de kostprijs van queries te berekenen<br/>C Een MongoDB installatie voldoet aan tegelijk aan C A en P<br/>D NoSQL databanken zijn altijd sneller dan relationele databanken<br/>E NoSQL voldoet aan meestal aan B, A, S en E<br/>F Een expliciete INNER JOIN geeft betere prestatie dan dezelfde conditie in de WHERE<br/>G Een goeie PostgreSQL installatie voldoet tegelijk aan C A en P<br/>H Een standaard MySQL installatie voldoet niet tegelijk aan C A en P<br/>I FKs (vreemde sleutels) zijn een vorm van noodzakelijke redundantie<br/>J PostgreSQL is een zuiver relationele databank<br/>K PGAdmin is een beperkte relationele databank</p> <p>Dit is geen sql vraag, maar een meerkeuze vraag. Je antwoord met de letters die van toepassing zijn. (2p)</p> <p>bv</p> <p>BDE</p> <p>Opm:<br/>* Geldige oplossingen bestaan uit hoofdletters zonder gebruik van spaties of andere leestekens, alfabetisch gesorteerd.<br/>* Voorbeelden van ONGELDIGE oplossingen: BA, b a, 'AB', AB(spatie) .</p> <p>PS: Het is normaal dat sqldropbox een fout geeft, je kan dit negeren.</p> |  |
|------|---------------------------------------|---|--|

|      |                                       |   |   |
|------|---------------------------------------|---|---|
| 2023 | Databanken:<br>DB2<br>Examenvraag iii | <p>Welke van de volgende stellingen zijn JUIST?:</p> <p>A Relationale databanken zijn achterhaald.<br/>B Een MongoDB installatie voldoet aan tegelijk aan C A en P<br/>C Een goeie PostgreSQL installatie voldoet tegelijk aan C A en P<br/>D Een standaard MySQL installatie voldoet niet tegelijk aan C A en P<br/>E PG_BENCH kan je gebruiken om de kostprijs van queries te berekenen<br/>F PostgreSQL is een zuiver relationele databank<br/>G PGAdmin is een beperkte relationele databank<br/>H NoSQL databanken zijn altijd sneller dan relationele databanken<br/>I NoSQL voldoet aan meestal aan B, A, S en E<br/>J Een expliciete INNER JOIN geeft betere prestatie dan dezelfde conditie in de WHERE<br/>K Logische replicatie kan enkel voor de ganse databank</p> <p>Dit is geen sql vraag, maar een meerkeuze vraag. Je antwoord met de letters die van toepassing zijn. (2p)</p> <p>bv</p> <p>BDE</p> <p>Opm:<br/>* Geldige oplossingen bestaan uit hoofdletters zonder gebruik van spaties of andere leestekens, alfabetisch gesorteerd.<br/>* Voorbeelden van ONGELDIGE oplossingen: BA, b a, 'AB', AB(spatie) .</p> <p>PS: Het is normaal dat sqldropbox een fout geeft, je kan dit negeren.</p> |  |
|------|---------------------------------------|---|---|

ingelogd als lector: u0073950 - beheer | SQLDropbox is het resultaat van onderzoek en ontwikkeling (K. Beheydt en W. Bertels). De beschikbaarheid wordt niet gegarandeerd.