Jurgen Paapen

Stijn Pijman

Joost Oomen

Nick Sluiters

**--22INF1B--**

**Move with Friends**



Documentatie Module 2

Inhoudsopgave

[1. Samenvatting 3](#_Toc534451870)

[2. Ontwerpen voor relationele datapersistentie 4](#_Toc534451871)

[2.1. Verzamelen en valideren van functionele eisen voor een softwaresysteem. 4](#_Toc534451872)

[2.2. Ontwerpen en/of digitaliseren van een bedrijfsproces 5](#_Toc534451873)

[3. Realiseren van relationele datapersistentie 6](#_Toc534451874)

[3.1. Uitwerking van databasesysteem 6](#_Toc534451875)

[3.2. Bouwen van een eenvoudig softwaresysteem 6](#_Toc534451876)

[4. Beheren van relationele datapersistentie 7](#_Toc534451877)

[4.1. Opzetten en documenteren 7](#_Toc534451878)

[4.2. Beheerssysteem 7](#_Toc534451879)

[I. Bijlage: Sitemap 8](#_Toc534451880)

[II. Bijlage: Technisch ontwerp 9](#_Toc534451881)

[III. Bijlage: MySQL code om tabellen te maken 10](#_Toc534451882)

[IV. Bijlage: MySQL code om tabellen te ‘vullen’ 12](#_Toc534451883)

[V. Bijlage: MySQL code met QUERY om matechde gebruikers te tonen 13](#_Toc534451884)

# Samenvatting

Relationele datapersistentie, zo heet deze module. Dat begint bij het samenstellen van een plan van aanpak om de webomgeving dynamischer te maken en een databasesysteem vorm te geven waarom gebruikersgegevens, tijdstippen en activiteiten opgeslagen worden EN waarmee een koppeling is te maken tussen gebruikers met overeenkomstige tijdstippen en activiteitvoorkeuren.

Vervolgens moeten de wensen en eisen in een functioneel en technische ontwerp worden gebracht zodat de database ook daarwerkelijk gecodeerd kan worden. Door middel van relatistische gegevens wordt de database gevuld en getest of matchende paren ook daarwerklijk uit het systeem komen.

Dit alles is gedocumenteerd en bijgehouden via een online beheerssysteem.

# Ontwerpen voor relationele datapersistentie

Naar aanleiding van de vorige module zijn we als groep verder ingegaan op de wensen die onze persona’s zouden kunnen hebben mbt bewegen en sociaal samenkomen en wat dat voor onze website zou betekenen. Deze eisen en wensen zijn verwerkt in een databasemodel (o.a. ERD en technisch ontwerp).

## Verzamelen en valideren van functionele eisen voor een softwaresysteem.

Gebruikers dienen aan elkaar gekoppeld te worden op basis van vergelijkbare wensen mbt activiteiten, tijdstippen en voorkeur voor de sexe waarmee men wil kunnen afspreken (en in de toekomst wellicht de GPS locatie). Om deze gegevens te verzamelen dient er gebruik gemaakt te worden van een databasysteem. Dit systeem dient de gebruikersgegevens op te slaan, de voorkeuren mbt activiteiten en tijdstippen bij te houden per gebruiker EN de koppeling te kunnen maken tussen gebruikers met overeenkomstige voorkeuren.

Datgene wat de gebruiker zoekt en wat MoveWithFriends dat levert is weergegeven in een User-Case-Diagram (hieronder).

O 
Klant 
Move With Friends (applicatie) 
Registreren 
Verifieer wachtwoord 
Log in 
Toon fout in aanmelding 
Zoek 
sport/bewegingspartner 
Toets mogelijke sport/ 
bewegingspartner(s) 
Match met partner 
O 
Move With Friends 
organisatie 
Maak afspraak 
Chat 

Figuur 1: User-Case Diagram MoveWithFriends

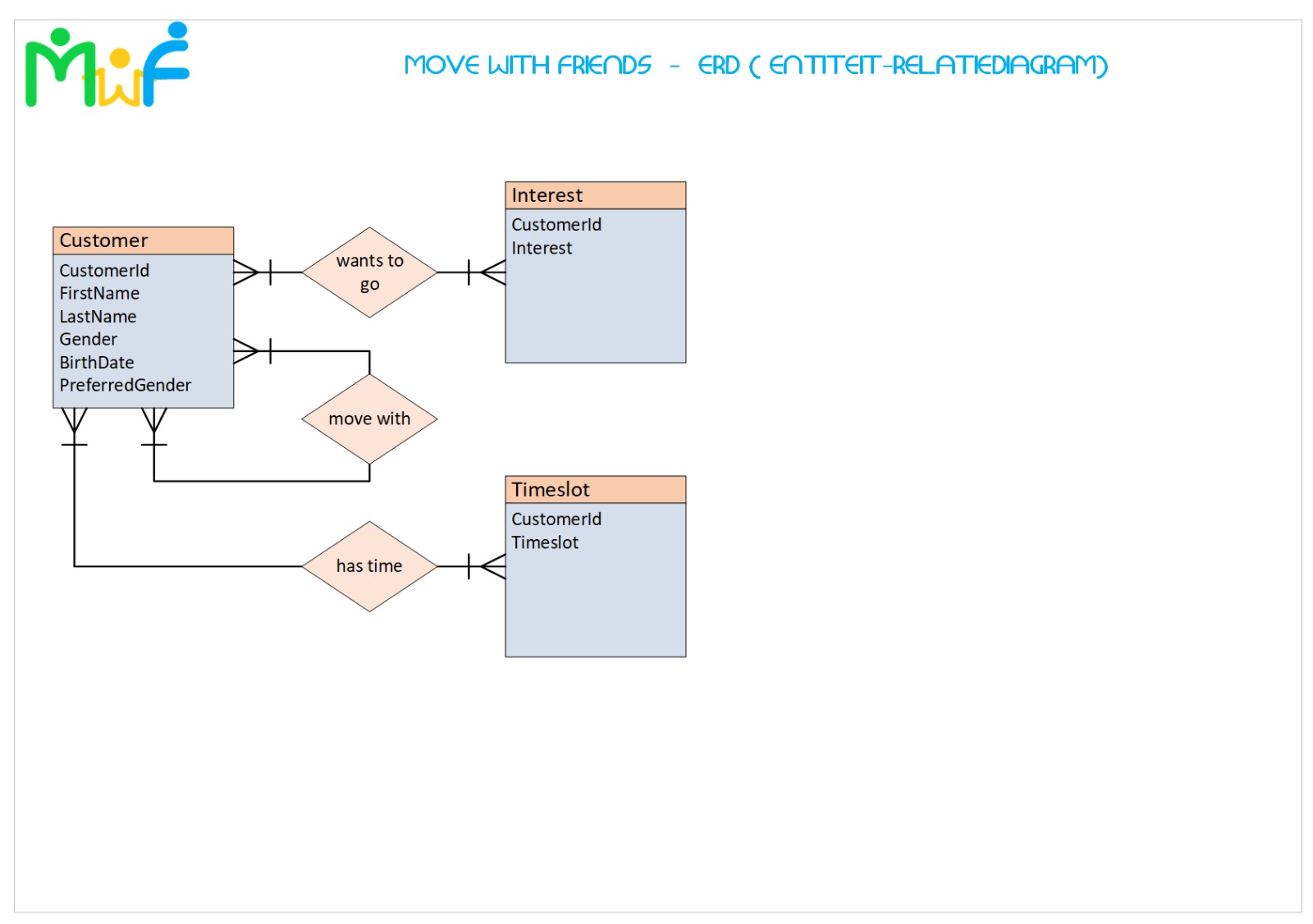
Om gebruikers in te kunnen laten registreren zou onze statische website een inschrijfmogelijkheid moeten hebben. Hiervoor is de sitemap van de oorsprokenelijke website aangepast (zie Bijlage I) Daarnaast zou de inlogmogelijkheid die op de statische website al aanwezig was ook functioneel moeten zijn en de gebruiker leiden naar een persoonlijke pagina. Dit inloggen zou plaats moeten vinden met verificatie van het wachtwoord dat gekoppeld is aan de gebruikersnaam. Dit wachtwoord dient versleuteld (via een hash) opgeslagen te worden in de database.

De gebruiker zou in zijn/haar persoonlijke webomgeving de voorkeuren moeten kunnen aangeven mbt de activiteiten en tijdsvoorkeuren. Nadat er een’match’ is met een andere gebruiker zou een mogelijkheid moeten zijn om tot een definitieve afspraak te komen via een chatfunctie. De overeenkomstige activiteit en tijdstip zouden bij beide gebruikers zichbaar moeten kunnen worden op de persoonlijke pagina. Deze laatste zaken voeren te ver voor de kennis en vaardigheden die we momenteel bezitten maar zullen in de toekomst zeker een onderdeel van MoveWithFriends omgeving vormen.

Om dit alles te kunnen laten functioneren zou de webpagina een koppeling moeten bevatten naar een database. Hierin zouden de inschrijfgegevens van de deelnemers en de keuze van de gebruikers mbt activiteit en beschikbaarheid, opgeslagen moeten worden. Bij het inloggen moeten gebruikersnaam en wachtwoord overeen komen om tot de persoonlijke pagina te komen.

De database zelf moet in staat zijn om gebruikers met overeenkomstige activiteitskeuzes EN overeenkomstige beschikbaarheid (qua tijdstip) te filteren en deze unieke koppelparen terug te kunnen geven. Dit laatste is wel een onderdeel van deze onderwijsperiode.

## Ontwerpen en/of digitaliseren van een bedrijfsproces

Het bedrijfproces dat is gekozen om te digitaliseren is het koppelen van overeenkomstige gebruikers (overeenkomt wat betreft activiteitenvoorkeur en beschikbare tijdsmoment). Om de entiteiten. Die bij dit proces weer te geven en de samenhang hiertussen ook weer te geven is er een ERD (Entiteiten Relatie Diagram)(zie hieronder).

Figuur 2: ERD van MoveWithFriends

Dit ERD bevat gebruikers gegevens zoals in hoofdstuk 1 aangegeven (Voornaam, achternaam, geslacht, gebruikersnaam, emailadres, telefoonnummer, wachtwoord, geboortedatum en voorkeurssexe om activiteiten mee uit te voeren). Interest bevat de verschillende activiteiten die gekozen kunnen worden en Timeslot bevat 21 dagdelen van de week (maandag t/m zondag, en ochtend, middag en avond).

Dit ERD hebben we vervolgens in een relationeel conceptueel model gezet en daarna een technisch ontwerp van gemaakt. Dit is te zien in bijlage II. Hierin is een koppeltabel te vinden waarin de koppeling tussen gebruiker en activiteiten en tijdslots plaatsvindt. Uit deze koppeltabel kan vervolgens overeenkomstige gebruikers gematched worden.

# Realiseren van relationele datapersistentie

Zoals in de vorige hoofdstuk aangegeven zijn er een aantal eisen toegevoegd aan het bestaande product. De website is aangepast zodat een extra tabblad voor het inschrijven aanwezig is. De globale opbouw van de website blijft vergelijkbaar met het wireframe dat we oorspronkelijk hadden gebruikt.   
De inschrijfpagina van de website omvat een formulier met daarin de mogelijkheid tot het invullen van een voornaam, achternaam, gebruikersnaam, emailadres, wachtwoord, geslacht en voorkeur voor sexe om de voorkeursactiviteiten uitmee te voeren.

Het inlogicoon heeft nu een invliegend inlogscherm dat ook functioneert. Als er ingelogd wordt komt men op een persoonlijke pagina en vanaf dan is inloggen niet meer nodig. Inlog blijft tijdens de actieve sessie bestaan. En het inlogicoon functioneert op de persoonlijke pagina als uitlogbutton.  
Op de persoonlijke pagina is de algemene informatie over Move with Friends weggehaald en komt men op een pagina waar het mogelijk is om de activiteiten te kiezen EN de tijdsmomenten (per dagdeel) waarop men beschikbaar is voor deze activiteiten (dit laatste is momenteel nog niet functioneel omdat ons de kennis (nog) ontbreekt dit werkzaam te krijgen.

## Uitwerking van databasesysteem

Er is voor het opslaan van de gegevens een database aangemaakt. Het ERD voor deze database is terug te vinden in hoofdstuk 1 (Figuur 2). Hierin zijn 3 entiteiten terug te vinden, de gebruiker, de activiteiten en de dagdelen. De laatste twee zijn door middel van een veel op veel relatie met de gebruiker verbonden. Dit is terug te zien in bijlage II waarin het technisch ontwerp terug te vinden is.

Om deze database vanuit een functioneel ontwerp te maken hebben we een SQL code gebruikt om de noodzakelijke tabellen te maken in MySQL. De SQL DDL voor het tot stand komen van de database staat in bijlage III (invoeren van de tabellen) en bijlage IV (‘vullen’ van de tabellen met data).

Het vullen van de database gebeurt normaal gesproken via een invoer vanuit de webomgeving. De gebruikersgegevens kunnen ook via de website ([www.movewithfriends.nl](http://www.movewithfriends.nl)) ingevoerd worden, maar de voorkeuren voor tijdstip en activiteiten koppelen aan de website voert boven de scope van deze module en hebben we dus niet toe kunnen passen.

## Bouwen van een eenvoudig softwaresysteem

Daarnaast moeten de gegevens uit de database gekoppeld kunnen worden. Gebruikers met dezelfde interesses en dezelfde tijdsmomenten in de week dienen aan elkaar gekoppeld te worden. De QUERY’s (SQL DML) die hiervoor gebruikt worden staan in bijlage V.

Deze QUERY vormt eerst een koppeltabel waarin de gegevens van de gebruikers gekoppeld wordt aan de tijdstippen en voorkeursactiviteiten. Vervolgens vindt er voor elke gebruiker een check plaats of deze met een andere gebruiker overeenkomst wat betreft tijdstip en voorkeursactiviteit. Daarnaast moet de gebruiker ook met die gebruiker overeenkomen qua voorkeurssexe. Deze overeenkomst moet van twee kanten aanwezig zijn. Pas dan worden de paren zichtbaar gemaakt in een view in de PhPMyAdmin omgeving.

# Beheren van relationele datapersistentie

De codes voor zowel de website als de database dienen voor alle medewerkers van MoveWithFriends beschikbaar te zijn. Om veranderingen in codes met elkaar te kunnen delen zijn systemen gebruikt om elkaar up-to-date te houden over deze veranderingen.

## Opzetten en documenteren

Op het platform van Github (<https://github.com/MoveWithFriends>) is ook dit verantwoordingsdocument terug te vinden. Alle gebruikers kunnen op deze manier zien welke keuzes er in het proces zijn gemaakt en waarom deze zijn gemaakt.

## Beheerssysteem

Bij MoveWithFriends gebruiken we Github (<https://github.com/MoveWithFriends>) als medium om codes te delen. Hierop worden zowel de codes voor de website als de codes voor de database gedeeld. Alle aanpassingen aan de codes worden zichtbaar voor anderen en na goedkeuring komen ze in het document te staan. Op die manier blijft er controle over de veranderingen en is voor iedereen inzichtlijk wat de veranderingen zijn die plaats hebben gevonden. De werkelijke codes zijn op deze manier ook altijd voor iedereen beschikbaar.

1. **Bijlage: Sitemap**

ryove um-I fRE-nD5 - 51Temap 
W'th 
I nschriiwn 
MOW With Friends 
page 
Versie 2, O 
Log in / Logoff 

1. **Bijlage: Technisch ontwerp**



1. Bijlage: MySQL code om tabellen te maken

|  |
| --- |
| DROP TABLE IF EXISTS `timeslot`, `interest`, `users` |
|  | ; |
|  |  |
|  | CREATE TABLE `users` ( |
|  | CustomerID INT(10) NOT NULL PRIMARY KEY AUTO\_INCREMENT, |
|  | firstname VARCHAR(255) NOT NULL, |
|  | lastname VARCHAR(255) NOT NULL,  preposition VARCHAR(255) NOT NULL, |
|  | email VARCHAR(255) NOT NULL, |
|  | username VARCHAR(255) NOT NULL, |
|  | phonenumber VARCHAR(20) NOT NULL, |
|  | gender VARCHAR(6) NOT NULL,  password VARCHAR(255) NOT NULL, |
|  | birthdate DATE NOT NULL, |
|  | preferredgender VARCHAR(10) NOT NULL) |
|  |  |
|  | ; |
|  |  |
|  |  |
|  | CREATE TABLE `interest` ( |
|  | CustomerIDinterest INT(10), |
|  | Boksen BOOLEAN, |
|  | Fitness BOOLEAN, |
|  | Hardlopen BOOLEAN, |
|  | Tennis BOOLEAN, |
|  | Squash BOOLEAN, |
|  | Wandelen BOOLEAN, |
|  | Wielrennen BOOLEAN, |
|  | Zwemmen BOOLEAN, |
|  |  |
|  | CONSTRAINT |
|  | PRIMARY KEY InterestPK (CustomerIDinterest, Boksen), |
|  |  |
|  | CONSTRAINT |
|  | FOREIGN KEY InterestFK (CustomerIDinterest) |
|  | REFERENCES users (CustomerID)) |
|  | ; |
|  |  |
|  | CREATE TABLE `timeslot` ( |
|  | CustomerIDtimeslot INT(10), |
|  | Timeslot1 BOOLEAN, |
|  | Timeslot2 BOOLEAN, |
|  | Timeslot3 BOOLEAN, |
|  | Timeslot4 BOOLEAN, |
|  | Timeslot5 BOOLEAN, |
|  | Timeslot6 BOOLEAN, |
|  | Timeslot7 BOOLEAN, |
|  | Timeslot8 BOOLEAN, |
|  | Timeslot9 BOOLEAN, |
|  | Timeslot10 BOOLEAN, |
|  | Timeslot11 BOOLEAN, |
|  | Timeslot12 BOOLEAN, |
|  | Timeslot13 BOOLEAN, |
|  | Timeslot14 BOOLEAN, |
|  | Timeslot15 BOOLEAN, |
|  | Timeslot16 BOOLEAN, |
|  | Timeslot17 BOOLEAN, |
|  | Timeslot18 BOOLEAN, |
|  | Timeslot19 BOOLEAN, |
|  | Timeslot20 BOOLEAN, |
|  | Timeslot21 BOOLEAN, |
|  |  |
|  | CONSTRAINT |
|  | PRIMARY KEY TimeslotPK (CustomerIDtimeslot, Timeslot1), |
|  |  |
|  | CONSTRAINT |
|  | FOREIGN KEY TimeslotFK (CustomerIDtimeslot) |
|  | REFERENCES users (CustomerID)) |
|  | ; |

1. Bijlage: MySQL code om tabellen te ‘vullen’

|  |
| --- |
| INSERT INTO `users` |
|  | VALUES (1, 'Frans', 'Jadus', '0648470015', 'Fransie@jaja.nl', 'Fransie', 'man', '1993-07-09', 'man'); |
|  |  |
|  | INSERT INTO `users` |
|  | VALUES (2, 'Jurger', 'Papjes', '0612345678', 'Jurger@neenee.nl', 'BurgerJurger', 'man', '1980-08-12', 'man'); |
|  |  |
|  | INSERT INTO `users` |
|  | VALUES (3, 'Joost', 'Dinges', '0612545678', 'Joostje@Whoepo.nl', 'Joostje', 'man', '1960-05-11', 'man'); |
|  |  |
|  | INSERT INTO `users` |
|  | VALUES (4, 'Stijn', 'Ookiets', '0612545678', 'Stijntjenijntje@Whiepie.nl', 'StijntjeNijntje', 'man', '1988-07-11', 'man'); |
|  |  |
|  | INSERT INTO `interest` |
|  | VALUES (1, 1, 0, 0, 0, 0, 0, 0, 0); |
|  |  |
|  | INSERT INTO `interest` |
|  | VALUES (2, 1, 0, 0, 0, 0, 0, 0, 0); |
|  |  |
|  | INSERT INTO `interest` |
|  | VALUES (3, 0, 0, 1, 0, 0, 0, 0, 0); |
|  |  |
|  | INSERT INTO `interest` |
|  | VALUES (4, 0, 0, 0, 0, 1, 0, 0, 0); |
|  |  |
|  |  |
|  | INSERT INTO `timeslot` |
|  | VALUES (1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0); |
|  |  |
|  | INSERT INTO `timeslot` |
|  | VALUES (2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0); |
|  |  |
|  | INSERT INTO `timeslot` |
|  | VALUES (3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1); |
|  |  |
|  | INSERT INTO `timeslot` |
|  | VALUES (4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1); |

1. Bijlage: MySQL code met QUERY om matechte gebruikers te tonen

|  |  |
| --- | --- |
|  | DROP VIEW IF EXISTS `koppel`; |
|  | CREATE VIEW koppel AS |
|  |  |
|  | SELECT \* from users |
|  | JOIN interest ON users.CustomerID = interest.CustomerIDInterest |
|  | JOIN timeslot ON users.CustomerID = timeslot.CustomerIDtimeslot; |
|  |  |
|  |  |
|  | SELECT a.CustomerID, a.timeslot1, a.timeslot2, a.timeslot3, a.timeslot4, a.timeslot5, a.timeslot6, a.timeslot7, a.timeslot8, a.timeslot9, a.timeslot10, a.timeslot11, a.timeslot12, a.timeslot13, a.timeslot14, a.timeslot15, a.timeslot16, a.timeslot17, a.timeslot18, a.timeslot19, a.timeslot20, a.timeslot21, a.boksen, a.fitness, a.Hardlopen, a.Tennis, a.Squash, a.Wandelen, a.Wielrennen, a.Zwemmen, a.FirstName, b.FirstName FROM koppel a |
|  | JOIN koppel b |
|  | ON a.customerID <> b.CustomerID |
|  | AND (a.preferredgender = b.gender |
|  | OR a.preferredgender = 'beide') |
|  | AND (b.preferredgender = a.gender |
|  | OR b.preferredgender = 'beide') |
|  | AND (a.timeslot1 = 1 AND b.timeslot1 = 1 |
|  | OR a.timeslot2 = 1 AND b.timeslot2 = 1 |
|  | OR a.timeslot3 = 1 AND b.timeslot3 = 1 |
|  | OR a.timeslot4 = 1 AND b.timeslot4 = 1 |
|  | OR a.timeslot5 = 1 AND b.timeslot5 = 1 |
|  | OR a.timeslot6 = 1 AND b.timeslot6 = 1 |
|  | OR a.timeslot7 = 1 AND b.timeslot7 = 1 |
|  | OR a.timeslot8 = 1 AND b.timeslot8 = 1 |
|  | OR a.timeslot9 = 1 AND b.timeslot9 = 1 |
|  | OR a.timeslot10 = 1 AND b.timeslot10 = 1 |
|  | OR a.timeslot11 = 1 AND b.timeslot11 = 1 |
|  | OR a.timeslot12 = 1 AND b.timeslot12 = 1 |
|  | OR a.timeslot13 = 1 AND b.timeslot13 = 1 |
|  | OR a.timeslot14 = 1 AND b.timeslot14 = 1 |
|  | OR a.timeslot15 = 1 AND b.timeslot15 = 1 |
|  | OR a.timeslot16 = 1 AND b.timeslot16 = 1 |
|  | OR a.timeslot17 = 1 AND b.timeslot17 = 1 |
|  | OR a.timeslot18 = 1 AND b.timeslot18 = 1 |
|  | OR a.timeslot19 = 1 AND b.timeslot19 = 1 |
|  | OR a.timeslot20 = 1 AND b.timeslot20 = 1 |
|  | OR a.timeslot21 = 1 AND b.timeslot21 = 1) |
|  | AND (a.boksen = 1 AND b.boksen = 1 |
|  | OR a.fitness = 1 AND b.fitness = 1 |
|  | OR a.hardlopen = 1 AND b.hardlopen = 1 |
|  | OR a.tennis = 1 AND b.tennis = 1 |
|  | OR a.squash = 1 AND b.squash = 1 |
|  | OR a.wandelen = 1 AND b.wandelen = 1 |
|  | OR a.wielrennen = 1 AND b.wielrennen = 1 |
|  | OR a.zwemmen = 1 AND b.zwemmen = 1); |
|  |  |