5-11-2020

Jeffrey Meijer

B6 Opdrachten

Inhoud

[Transactions 2](#_Toc56504876)

[Wat is een transaction en wat kun je er mee doen? 2](#_Toc56504877)

[In wat voor situatie word een transaction gebruikt? 2](#_Toc56504878)

[Procedures 2](#_Toc56504879)

[Wat is een stored procedure en wat kun je er mee doen? 2](#_Toc56504880)

[In wat voor situatie word een procedure gebruikt? 2](#_Toc56504881)

[Is het nodig om zoveel inner joins aanwezig te hebben bij 2.2.2.3 – Moeilijke stored procedure? 2](#_Toc56504882)

[Hoe maak je een procedure aan en laat je deze callen in een transaction? 2](#_Toc56504883)

[Triggers & events – vragen 3](#_Toc56504884)

[Vraag 1. 3](#_Toc56504885)

[Vraag 2. 3](#_Toc56504886)

[Vraag 3. 3](#_Toc56504887)

[Vraag 4. 3](#_Toc56504888)

[Vraag 5. 3](#_Toc56504889)

[Vraag 6. 4](#_Toc56504890)

[telVolwassenenInsert 4](#_Toc56504891)

[telVolwassenenDelete 4](#_Toc56504892)

[telVolwassenenUpdate 4](#_Toc56504893)

[Vraag 7. 4](#_Toc56504894)

[Vraag 8. 4](#_Toc56504895)

[Triggers & events – opdrachten 4](#_Toc56504896)

[Opdracht 1. 4](#_Toc56504897)

[Opdracht 2. 5](#_Toc56504898)

[Opdracht 3. 5](#_Toc56504899)

[Stored functions 6](#_Toc56504900)

[Stored functions - vragen 6](#_Toc56504901)

# 

# Transactions

## Wat is een transaction en wat kun je er mee doen?

Een transaction is in feite gewoon een soort van back up of een extra beveiliging voor het geval een query fout gaat, als je perongeluk database entries verwijderd zonder een transactie kun je dit niet meer terug krijgen, met een transaction kun je het altijd nog terugzetten als het nog niet gecommit is (opgestuurd is zeg maar).

## In wat voor situatie word een transaction gebruikt?

De transaction zou in mijn ogen kunnen worden gebruikt voor verandering die nog wel eens mis zouden kunnen gaan, hiermee zorgen ze ervoor dat ze in geval van een error of iets dat er op lijkt het weer terug kunnen draaien zoals het was. Zo is er eigenlijk geen data verlies.

# Procedures

## Wat is een stored procedure en wat kun je er mee doen?

Een stored procedure is in feite een function zoals in php voor SQL code.

Je defined een procedure om een bepaalde SQL code te runnen (Of meerdere) en deze stored hij dan, deze zou je dan kunnen callen en dan runt hij de SQL code die in de procedure (function) zit.

Dit kan handig zijn om bijvoorbeeld klantgegevens op te halen, zo kun je met 1 procedure een klant zijn gegevens ophalen, zijn bestelgeschiedenis en misschien nog meer tabellen met een inner join.

Zo hoef je niet de heletijd de hele SQL query in te typen maar kun je gewoon een procedure runnen die het voor je doet.

## In wat voor situatie word een procedure gebruikt?

Voor herhalende queries, bijvoorbeeld het selecten van iedereen die heeft betaald of de hoeveelheid hiervan tellen, alle queries die je vaker wil/kan gebruiken kun je in een procedure zetten zodat deze makkelijk te callen is.

## Is het nodig om zoveel inner joins aanwezig te hebben bij 2.2.2.3 – Moeilijke stored procedure?

Ik denk zelf van wel, je moet data uit boeking,reis en bestemming tabellen halen, om deze data bij elkaar te krijgen zou je ze allemaal zelf moeten toevoegen (inner joinen dus).

Wel is het pluspunt ervan dat je dit maar 1 keer hoeft te doen want hierna kun je de procedure gewoon callen 😊

## Hoe maak je een procedure aan en laat je deze callen in een transaction?

Om een procedure aan te maken kun je het volgende doen (dit is om de klantbestemming te pakken)

CREATE PROCEDURE klantBestemming()

SELECT Naam, Woonplaats, bestemming.Plaats, bestemming.Land, bestemming.Werelddeel FROM klant

INNER JOIN boeking ON klant.Klantnr = boeking.Klantnr

INNER JOIN reis ON boeking.Reisnr = reis.Reisnr

INNER JOIN bestemming ON reis.Bestemmingcode = bestemming.Bestemmingcode

Nu om de procedure te callen in een transaction:

START TRANSACTION;

CALL klantBestemming();

ROLLBACK; (Kan ook COMMIT; zijn, ligt eraan wat de procedure doet)

# Triggers & events – vragen

## Vraag 1.

Bij elke insert in ‘boeking’ word er de aantal volwassenen gepakt uit boekingen, de rijen en pakt hier de eerste rij van uit ‘klant\_statistieken’ en pakt hij de datum van vandaag.

Hierna kijkt hij of de aantal rijen 0 of 1 is:

* Als de aantalrijen 0 is dan gebeurd het volgende:
  + De ‘klant\_statistieken’ krijgt een nieuwe rij toegevoegd met de nodige data.
* Als de aantalrijen 1 is dan gebeurd het volgende:
  + De aantalvolwassenen word geüpdatet met de nieuwe value en ook word de ‘laatste\_update’ kolom geüpdatet met de datum van vandaag.

Hierna is de trigger klaar met zijn taak.

## Vraag 2.

Na het inserten van een record is het volgende gebeurd:

In de ‘klant\_statistieken’ tabel is een rij aangemaakt met daarin het aantal volwassenen, dit zijn er 57.

## Vraag 3.

Na het wijzigen van de record is er in de tabel ‘klant\_statistieken’ het aantal volwassenen automatisch veranderd naar 59.

## Vraag 4.

Na het verwijderen van de nieuwe record is er in de tabel ‘klant\_statistieken’ het aantal volwassenen die in de record stonden eraf gehaald.

## Vraag 5.

Ik denk dat de delimiter $$ ervoor zorgt dat je niet zomaar de trigger stopt, elke lijn moet namelijk eindigen op ; en de delimiter staat normaal gesproken op ;

Als je dus een lijn zou eindigen dan eindig je vanaf die ; ook de hele trigger en snapt mysql er niks van.

## Vraag 6.

### telVolwassenenInsert

Op het moment dat er een nieuwe boeking word aangemaakt telt hij alle volwassenen op en zet dit in ‘klant\_statistieken’

### telVolwassenenDelete

Op het moment dat er een boeking word verwijderd telt hij alle volwassenen op en zet dit in ‘klant\_statistieken’

### telVolwassenenUpdate

Op het moment dat er een boeking word geüpdatet telt hij alle volwassenen opnieuw op en zet dit in ‘klant\_statistieken’

## Vraag 7.

Je kunt gebruik maken van een trigger & event voor (zoals gezien is) het optellen van hoeveel volwassenen er in totaal zijn. Zo kun je gemakkelijk het totaal aantal volwassenen zien en ophalen mocht dit nodig zijn.

## Vraag 8.

Een trigger & event heeft zoals de naam al zegt een trigger die ervoor zorgt dat een event gebeurd.

Als je als trigger hebt om iets te doen als er iets word toegevoegd in bijvoorbeeld ‘bestemmingen’ dan kun je bijvoorbeeld als event hebben om het totaal aantal bestemmingen te pakken en deze ergens op te slaan.

Zo update hij elke keer het totaal aantal bestemmingen als er 1 of meerdere bestemmingen worden toegevoegd.

# Triggers & events – opdrachten

## Opdracht 1.

|  |
| --- |
| DELIMITER $$  **CREATE** **TRIGGER** `telKinderenInsert`  AFTER **INSERT** **ON** `boeking`  **FOR** **EACH** ROW  BEGIN  **SET** @aantalKinderen = (**SELECT** **SUM**(Aantal\_kids) **FROM** boeking);  **SET** @aantalRijen = (**SELECT** **COUNT**(**IFNULL**(Aantal\_kinderen, **1**)) **FROM** klant\_statistieken);  **SET** @datum = (**SELECT** **CURDATE**());  **IF** @aantalRijen = **0** **THEN**  **INSERT** **INTO** **klant\_statistieken** (Aantal\_kinderen) **VALUES** (@aantalKinderen);  **ELSEIF** @aantalRijen = **1** **THEN**  **UPDATE** klant\_statistieken  **SET** Aantal\_kinderen = @aantalKinderen, Laatste\_update = @datum **WHERE** **1**;  END **IF**;  END$$  DELIMITER ; |

## Opdracht 2.

|  |
| --- |
| DELIMITER $$  **CREATE** **TRIGGER** `telBetaaldeKlanten`  AFTER **UPDATE** **ON** `boeking`  **FOR** **EACH** ROW  BEGIN  **SET** @aantalBetaald = (**SELECT** **COUNT**(status) **FROM** boeking **WHERE** status = 'Betaald');  **SET** @aantalRijen = (**SELECT** **COUNT**(**IFNULL**(Aantal\_klanten\_betaald, **1**)) **FROM** klant\_statistieken);  **SET** @datum = (**SELECT** **CURDATE**());  **IF** @aantalRijen = **0** **THEN**  **INSERT** **INTO** **klant\_statistieken** (Aantal\_klanten\_betaald) **VALUES** (@aantalBetaald);  **ELSEIF** @aantalRijen = **1** **THEN**  **UPDATE** klant\_statistieken  **SET** Aantal\_klanten\_betaald = @aantalBetaald, Laatste\_update = @datum **WHERE** **1**;  END **IF**;  END$$  DELIMITER ; |

## Opdracht 3.

|  |
| --- |
| DELIMITER $$  **CREATE** **TRIGGER** `telOnbetaaldeklanten`  AFTER **DELETE** **ON** `boeking`  **FOR** **EACH** ROW  BEGIN  **SET** @aantalOnbetaald = (**SELECT** **count**(status) **FROM** boeking **WHERE** status != 'Betaald');  **SET** @aantalRijen = (**SELECT** **COUNT**(**IFNULL**(Aantal\_klanten\_niet\_betaald, **1**)) **FROM** klant\_statistieken);  **SET** @datum = (**SELECT** **CURDATE**());  **IF** @aantalRijen = **0** **THEN**  **INSERT** **INTO** **klant\_statistieken** (Aantal\_klanten\_niet\_betaald) **VALUES** (@aantalOnbetaald);  **ELSEIF** @aantalRijen = **1** **THEN**  **UPDATE** klant\_statistieken  **SET** Aantal\_klanten\_niet\_betaald = @aantalOnbetaald, Laatste\_update = @datum **WHERE** **1**;  END **IF**;  END$$  DELIMITER ; |

# Stored functions

## Stored functions - vragen