Wat is een transaction

Een soort van query function die je terug kan draaien.   
mocht er iets fout gaan

Wanneer gebruik je een transaction

Je gebruikt het als een backup als er iets fout gaat dat je het dan kan verbeteren.

Wat is het verschil dus Commit en Rollback

Commit is dan is het set dus kan niet meer terug en rollback gaat die terug naar de laatste savepoint.

Hoe gebruik je savepoints in een transaction

Save points gebruik in je combinatie met roll back als je wilt opslaan tot dit punt.   
Dan gebruik je de save point en dan als je terug wilt naar dat punt gebruik je rollback en dan gaat  
die naar de laatste save point.

Wat is een stored procedure

Het is een functie die een query execute

Waneer gebruik je een stored procedure

Als je vaker een query gebruikt.

Waarom zijn er zo veel innerjoins bij 2.2.2.3

Het heeft informatie uit een andere table nodig maar daarvoor is het nog eerst andere informatie uit andere table nodig.

Hier mee kan je alle betaalde boekingen zien en waar ze heen gaan

CREATE PROCEDURE betaalde()

SELECT bestemming.Plaats, bestemming.Land, status FROM boeking

INNER JOIN reis ON boeking.Reisnr = reis.Reisnr

INNER JOIN bestemming ON reis.Bestemmingcode = bestemming.Bestemmingcode

WHERE status = 'Betaald'

Triggers & events – vragen

1.

Na dat iets gedelete word dan triggerd dit

Pakt eerst het aantal volwassenen

Dan het aantal rijen dat er zijn als het 0 is dan returned die 1

Dan pakt die de current datum

Als er niks in zit in de table dan maakt die een row als er wel een row is dan update die naar de nieuwe informatie. Dus aantal volwassen, aantal rijen , current datum

2.

DELIMITER $$

CREATE TRIGGER `telBetaaldUpdate`

AFTER UPDATE ON `boeking`

FOR EACH ROW

BEGIN

SET @aantalBetaald = (SELECT SUM(status) FROM boeking WHERE status = 'Betaald');

SET @aantalRijen = (SELECT COUNT(IFNULL(Aantal\_volwassenen, 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 ;

3.

Kan het niet werkend krijgen krijg een rare error die zegt dat de table niet bestaat. En als toen probeerde ik de trigger tedeleten met DROP TRIGGER naam maar dan bestaat die als nog en als je hem dan wilt deleten alweer zegt die dat die er niet is…

Maar ik ga er van uit dat het na dat boekingen word geupdate dit doet:

Zet de nieuwe aantal volwassenen in de statistieken table en de update datum.

4.

Wat ik er van uit ga dat het gaat doen :

Hetzelfde als bij vraag 6 ongeveer maar dan is de trigger on DELETE

5.

Het is een soort van grens volgens mij die dingen gescheiden houd

6.

het telt het aantal volwassenen

telt het aantal rows

haalt de current date op

checkt dat er een row is in klant\_statistieken

zo ja dan update die row

zo nee dan insert die een nieuwe row

7.

Als je bijvoorbeeld de totale statistieken bij wilt houden.

8.

Trigger event is een function die getrigger word bij een SELECT,UPDATE,DELETE

Trigger & Events – Opdracht

1.

DELIMITER $$

CREATE TRIGGER ` telKinderenUpdate`

AFTER INSERT ON `boeking`

FOR EACH ROW

BEGIN

SET @aantalKinderen = (SELECT SUM(Aantal\_kinderen) FROM boeking);

SET @aantalRijen = (SELECT COUNT(IFNULL(Aantal\_volwassenen, 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 ;

2.

DELIMITER $$

CREATE TRIGGER ` telBetaaldUpdate`

AFTER UPDATE ON `boeking`

FOR EACH ROW

BEGIN

SET @aantalBetaald = (SELECT SUM(status) where FROM boeking WHERE status = 'Betaald');

SET @aantalRijen = (SELECT COUNT(IFNULL(Aantal\_volwassenen, 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 ;

3.

DELIMITER $$

CREATE TRIGGER ` telNietBetaaldUpdate`

AFTER DELETE ON `boeking`

FOR EACH ROW

BEGIN

SET @aantalNietBetaald = (SELECT SUM(status) where FROM boeking WHERE status = 'Niet betaald!');

SET @aantalRijen = (SELECT COUNT(IFNULL(Aantal\_volwassenen, 1)) FROM klant\_statistieken);

SET @datum = (SELECT CURDATE());

IF @aantalRijen = 0 THEN

INSERT INTO klant\_statistieken (Aantal\_klanten\_niet\_betaald) VALUES (@aantalNietBetaald);

ELSEIF @aantalRijen = 1 THEN

UPDATE klant\_statistieken

SET Aantal\_klanten\_niet\_betaald = @aantalNietBetaald, Laatste\_update = @datum WHERE 1;

END IF;

END$$

DELIMITER ;

Stored functions - vragen

1.

Deterministisch geeft altijd dezelfde waarde terug

Niet deterministisch geeft een waarde terug die van de ingevoerde waarde komt

2.

Geeft de type return dus een int of een varchar en mee r

3.

Een stored function returned 1 value ??? en de andere kan meerdere.  
Ook kan een stored function geen stored procedure aanroepen maar een stored procedure kan wel ween stored function aanroepen.

Stored functions – opdracht

Opdracht 1

DELIMITER $$

CREATE FUNCTION berekenBelasting(bedrag int(7), belasting int(7))

RETURNS int(7) DETERMINISTIC

BEGIN

SET @nieuwBedrag = bedrag \* (1 - belasting / 100);

RETURN @nieuwBedrag;

END $$

DELIMITER ;

Opdracht 2

DELIMITER $$

CREATE FUNCTION newPrice(bedrag int(7), belasting int(7))

RETURNS int(7) DETERMINISTIC

BEGIN

SET @nieuwBedrag = bedrag \* (1 + belasting / 100);

RETURN @nieuwBedrag;

END $$

DELIMITER ;

Views

Hoe zou je een view in je eigen woorden omschrijven?

Het is table die niet zelf dingen opslaat maar uit andere tables dingen haalt

Wat is het voornaamste doel van een view ?

Beveiliging en dingen makkelijker maken zo dat je minder selects hoeft te gebruiken

Kan je een stored function aanroepen in een view

Ja dat kan

Views – Opdrachten

CREATE VIEW landReizen AS

SELECT bestemming.Plaats, bestemming.Land,Aantal\_volwassenen,Aantal\_kids,Betaald\_bedrag FROM boeking

INNER JOIN reis ON boeking.Reisnr = reis.Reisnr

INNER JOIN bestemming ON reis.Bestemmingcode = bestemming.Bestemmingcode

Eindopdracht

Opdracht A

START TRANSACTION;

SAVEPOINT beginInsert

INSERT INTO `boeking`(`Klantnr`, `Reisnr`, `Boekdatum`,`Aantal\_volwassenen`, `Aantal\_kids`, `Betaald\_bedrag`, `status`)

VALUES (7,15,CURDATE(),2,2,500.00, "");

SAVEPOINT FirstInsert;

UPDATE boeking SET status = CASE

WHEN Betaald\_bedrag > 0 THEN "Betaald"

WHEN Betaald\_bedrag <= 0 THEN "Niet betaald!"

ELSE "NIET BETAALD!"

END

WHERE 1;

SAVEPOINT SecondInsert;

CALL boekingen();

INSERT INTO `boeking`(`Klantnr`, `Reisnr`, `Boekdatum`,`Aantal\_volwassenen`, `Aantal\_kids`, `Betaald\_bedrag`, `status`)

VALUES (7,15,CURDATE(),2,2,500.00, "");

SAVEPOINT ThirdInsert

ROLLBACK TRANSACTION SecondInsert

CALL boekingen();

Opdracht B

CREATE PROCEDURE OpdrachB()

BEGIN

SELECT Aantal\_volwassenen,Aantal\_kids FROM `boeking` WHERE Boekingnr = 1;

SELECT Aantal\_volwassenen INTO @Aantal\_volwassenen

FROM boeking WHERE Boekingnr = 1;

SELECT Aantal\_kids INTO @Aantal\_kids

FROM boeking WHERE Boekingnr = 1;

SELECT Betaald\_bedrag INTO @Betaald\_bedrag

FROM boeking WHERE Boekingnr = 1;

SELECT AddUpp(@Aantal\_volwassenen,@Aantal\_kids)

AS aantal\_mensen INTO @total;

SELECT pricePer(@Betaald\_bedrag,@total) AS prijs\_per\_persoon;

END