트리거(trigger) - 정보가 수정됐을 때 연결된 테이블에서 참조하는 값을 함께 변경한다

before - 트리거를 작동한 뒤 쿼리문을 실행하게 함

after - 쿼리문을 실행한 뒤 트리거가 작동하게 함

instead of - 트리거를 원래 문장 대신 수행하게 함

when - 트리거를 시작하는 조건식을 지정

문법

DELIMITER //

CREATE TRIGGER 트리거명

BEFORE|AFTER INSERT|UPDATE|DELETE ON 테이블명

[REFERENCING NEW ROW AS n, OLD ROW AS O]

FOR EACH ROW -> 실행문이 여러개일 때 하나씩 실행한다(향상된 for문과 비슷)

BEGIN

...

END //

DELIMITER ;

트리거는 재귀호출로 자신을 호출하여 자신을 수정하는 방법으로는 사용할 수 없다

-> 프로시저 사용

green DB에 테이블 생성

CREATE TABLE `class` (

`no` int(11) NOT NULL,

`name` varchar(45) DEFAULT NULL,

`total` int(11) NOT NULL DEFAULT '0',

PRIMARY KEY (`no`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8

course 테이블 생성

CREATE TABLE `course` (

`no` int(11) NOT NULL,

`class\_no` int(11) NOT NULL,

`student\_no` int(11) NOT NULL,

KEY `class\_no\_idx` (`no`),

CONSTRAINT `class\_no` FOREIGN KEY (`no`) REFERENCES `class` (`no`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB DEFAULT CHARSET=utf8

트리거 생성

use green; DROP TRIGGER IF EXISTS input\_course;

DELIMITER //

CREATE TRIGGER input\_course AFTER INSERT ON course

FOR EACH ROW

BEGIN

declare \_total int default 0;

set \_total = (select count(\*) from course where new.class\_no = class\_no);

update class

set total = \_total

where no = new.class\_no;

END//

DELIMITER ;

show triggers;

테스트

insert into class(no,name,total) values(1,'컴퓨터공학',0);

insert into course(no,class\_no,student\_no) values(1, 1, 2019160135);

SELECT \* FROM green.class;

Gym 트리거 생성

course 테이블에 insert시

-> 1.프로그램 no가 일치하는 프로그램 테이블의 program\_total을 증가시킨다

-> 2. 멤버 id가 일치하는 멤버 테이블의 member\_course\_no에 course\_no을 입력한다

use gym;

drop trigger if exists add\_course;

delimiter //

create trigger add\_course after insert on course

for each row

begin

declare \_program\_total int default 0;

declare \_member\_course\_no int default 0;

set \_program\_total = (select count(\*) from course where new.course\_program\_no = course\_program\_no);

set \_member\_course\_no = new.course\_no;

update program

set program\_total = \_program\_total

where program\_no = new.course\_program\_no;

update member

set member\_course\_no = \_member\_course\_no

where member\_id = new.course\_member\_id;

end //

delimiter ;

show triggers;

예제) update 트리거를 통해 old를 사용

course에서 give\_up의 값을 변경하면 class\_no이 일치하는 class테이블의 total이 변경되는 트리거

use green;

DROP TRIGGER IF EXISTS update\_course;

DELIMITER //

CREATE TRIGGER update\_course AFTER update ON course

FOR EACH ROW

BEGIN

-- 수강 신청했다가 포기를 하는 경우 -> give\_up 테이블의 값이 N이었다가 Y로 바뀌는 경우

if old.give\_up = 'N' and new.give\_up = 'Y' then

update class

set total = total-1

where no = new.class\_no;

end if;

-- 수강 포기를 철회하는 경우 -> give\_up 테이블의 값이 Y였다가 N으로 바뀌는 경우

if old.give\_up = 'Y' and new.give\_up = 'N' then

update class

set total = total+1

where no = new.class\_no;

end if;

END//

DELIMITER ;

show triggers;

예제)수강 상태가 N인 수강 목록이 삭제되면 class\_no가 일치하는 class테이블의 total을 변경하는 트리거

drop trigger if exists delete\_course;

delimiter //

create trigger delete\_course after delete on course

for each row

begin

if old.give\_up = 'N' then

update class

set total = total-1

where old.class\_no = no;

end if;

end //

delimiter ;