**пСАНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Институт информационных технологий и управления**

**Кафедра компьютерных систем и программных технологий**

**ОТЧЕТ**

**по лабораторной работе №6**

**«SQL-программирование: триггеры, вызовы процедур»**

**Информационное обеспечение систем управления**

Студент гр. 43501/1 Матлаш И.Г.

Преподаватель Мяснов А.В.

Санкт-Петербург

2015

# 1. Цель работы

Ознакомиться с возможностями реализации более сложной обработки данных на стороне сервера с помощью хранимых процедур и триггеров.

# 2. Программа работы

1. Создать два триггера: один триггер для автоматического заполнения ключевого поля, второй триггер для контроля целостности данных в подчиненной таблице при удалении/изменении записей в главной таблице

2. Создать триггер в соответствии с индивидуальным заданием, полученным у преподавателя

3. Создать триггер в соответствии с индивидуальным заданием, вызывающий хранимую процедуру

# 3. Общие примеры триггеров

Триггер привязан к конкретной таблице и выполняется до или после заданного типа операции над ней. Если триггеров у одной таблицы несколько, порядок их выполнения можно задавать с помощью position.

## 3.1 Триггер для автоматического заполнения ключевого поля таблицы person.

Ранее был создан генератор идентификаторов для таблицы:

create sequence gen\_person\_id;

ALTER SEQUENCE gen\_person\_id RESTART WITH 9999;

Триггер:

set term ^ ;

create or alter trigger tr\_gen\_person\_id for person

active before insert position 0

as

begin

if (new.id is null) then

begin

new.id = gen\_id(gen\_person\_id, 1);

end

end ^

set term ; ^

Выполняется проверка на присутствие ключевого поля в новой записи, если условие не выполнено, то перед добавлением новой записи в ней заполняется ключевое поле.

Пример использования:

select count(\*) from person;

COUNT

9995

insert into person values (null,'Alex','89119765365','SPB,Kirochnaya 2-10',null);

select count(\*) from person;

COUNT

9996

select first 1 \* from person order by -id;

ID NAME TEL ADRESS MONEY

10000 Alex 89119765365 SPB,Kirochnaya 2-10

Триггер сработал, запись с новым id добавлена.

## 3.2 Триггер для контроля целостности данных в подчиненной таблице при удалении/изменении записей в главной таблице.

Контролируется удаление записей из таблицы car (объявлений), которое должно вызывать удаление статистики просмотра этих объявлений и истории пополнения счёта объявлений из соответствующей таблицы.

Попробуем удалить объявление до добавления триггера к таблице:

delete from car where id=43;

violation of FOREIGN KEY constraint "INTEG\_50" on table "VIEWS".

Foreign key references are present for the record.

Problematic key value is ("ID" = 43).

Триггер:

set term ^ ;

create or alter trigger tr\_del\_car for car

active before delete position 0

as begin

delete from views where car = old.id;

delete from money where car\_id = old.id;

end ^

set term ; ^

Без триггера эту операцию вообще невозможно было бы совершить, так как она нарушает контроль ссылочной целостности.

delete from car where id=43;

select \* from views where car = 43;

В результате удаляется и объявление и связанные с ним данные.

# 4. Индивидуальное задание

## 4.1 При создании объявления проверять наличие аналогичного объявления у этого пользователя, если находится - не создавать.

Исключение:

create or alter exception CAR\_EXIST 'Такое объявление уже существует.';

Триггер:

set term ^ ;

create or alter trigger tr\_check\_car\_unique for car

active before insert position 0

as

begin

if( exists (select id from car where body=new.body and condition=new.condition and engine=new.engine and model=new.model and transmission=new.transmission and person=new.person and edition\_year = new.edition\_year and color=new.color)) then exception CAR\_EXIST;

end ^

set term ; ^

Создается исключение типа CAR\_EXIST.

Попробуем создать объявление, аналогичное объявлению с id=44:

select \* from car where id = 44;

ID PHOTO MODEL MILEAGE EDITION\_YEAR BODY ENGINE TRANSMISSION COLOR CONDITION REVIEW PRICE PERSON STATUS CREATE\_DATE DELETE\_DATE PAYED\_END PAYED

44 5 579714 1991 4 1 1 Silver 1 62216 1915 1 02.01.2015 24.03.2015 20

INSERT INTO CAR VALUES (10000, NULL, 5, 579714, 1991, 4, 1, 1, 'Silver', 1, NULL, 62216, 1915, 1, '2-JAN-2015', NULL, '24-MAR-2015', 20);

CAR\_EXIST.

Такое объявление уже существует.

At trigger 'TR\_CHECK\_CAR\_UNIQUE' line: 9, col: 5.

Операция не выполнена, объявление не добавлено.

## 4.2 При проведении оплаты за объявление - менять статус.

Процедура высчитывает и возвращает новое значение счёта в зависимости от текущей стоимости одного месяца (параметр - COST), а также возвращает количество месяцев на которое следует продлить объявление (параметр – DD).

Процедура:

SET TERM ^ ;

CREATE OR ALTER procedure MATH\_PAYED (

PAYED decimal(10,2),

COST decimal(10,2))

returns (

DD integer,

NEW\_PAYED decimal(10,2))

as

declare variable DIV decimal(10,2);

begin

div = :payed/:cost;

dd = FLOOR(div);

new\_payed = :payed-(dd\*:cost);

end

^

SET TERM ; ^

Триггер:

SET TERM ^ ;

CREATE OR ALTER trigger tr\_payed for car

active before update position 0

as

declare variable dd int;

begin

if (new.payed > old.payed) then

begin

execute procedure math\_payed (new.payed,10) RETURNING\_VALUES (dd,new.payed);

if (old.payed\_end is null) then new.payed\_end = DATEADD(:dd MONTH to CURRENT\_DATE);

else new.payed\_end = DATEADD( MONTH, :dd, old.payed\_end);

end

end

^

SET TERM ; ^

Проверка:

В триггере была задана стоимость одного месяца, равная 10.

select id,payed,payed\_end from car where id = 45;

ID PAYED PAYED\_END

45 0 09.01.2016

Попробуем заплатить за 2 месяца суммой, которая превышает цену двух месяцев, но меньше цены трех, чтобы проследить остаток на счёте.

update car set payed = payed+23.00 where id=45;

select id,payed,payed\_end from car where id = 45;

ID PAYED PAYED\_END

45 3 09.03.2016

Триггер срабатывает верно, а процедура производит верные расчёты. В итоге объявление продлевается, а остаток внесённой суммы записывается на счёт.

# 5. Вывод

В ходе данной работы были изучены триггеры. Триггер связан с определенной таблицей и вызывается автоматически до или после определенного действия по изменению записей в таблице. Триггеры – основной способ контроля целостности данных при удалении записи в одной из связанных таблиц. Триггер можно использовать для автоматического заполнения полей таблицы при добавлении записей или контроля проводимых операций. При этом триггеры усложняют работу БД и негативно влияют на производительность.