Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт по лабораторной работе**

**Дисциплина**: Базы данных

**Тема**: Триггеры, вызовы процедур

Выполнила студентка гр. 43501/4 Хрусталева М.С.

(подпись)

Руководитель Мяснов А.В.

(подпись)

“\_ ” 2015 г.

Санкт - Петербург

2015

**Цель работы:**

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

**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
4. Выложить скрипт с созданными сущностями в svn
5. Продемонстрировать результаты преподавателю

**Выполнение работы**

**1.**  Был создан триггер для автоматического заполнения ключевого поля таблицы животных. Для заполнения ключевого поля использовался генератор.

CREATE GENERATOR forPrimaryKey;

SET GENERATOR forPrimaryKey TO animals.idanimals;

drop trigger animalsprimarykey;

create trigger animalsPrimaryKey for animals before insert

as

begin

NEW.idanimals = GEN\_ID(forPrimaryKey, 1);

end;

**2.** Был создан триггер для контроля целостности данных в подчиненной таблице отрядов животных при удалении/изменении записей в главной таблице классов животных.

create exception wrongDeleting 'Deleting animal class with references';

drop trigger deleteAnimalClass;

create trigger deleteAnimalClass for animalclass before delete or update

as

begin

if (OLD.classid in (select detachment.classid from detachment)) then

exception wrongDeleting;

end;

**3.** В соответствии с индивидуальным заданием был создан триггер, который при добавлении записей в таблицу потребления проверяет наличие в данной поставке потребляемого продукта и непревышение количества (с учетом уже существующих потреблений по данной поставке).

create exception amountError 'too much eat';

create exception deliverNotFound 'This product has not delivered in this delivery';

drop trigger eatInsert;

create trigger eatInsert for eat before insert

as

declare variable totalAmount numeric(9,0);

declare variable totalUsed numeric(9,0);

begin

for select linkprod.amount from linkprod

where linkprod.prodid = new.prodid and

linkprod.deliver\_id = new.deliv\_id

into :totalAmount do begin

select SUM(eat.amount) from eat where

eat.deliv\_id = new.deliv\_id and

eat.prodid = new.prodid

into :totalUsed;

if (totalUsed is null) then totalUsed = 0;

if (totalUsed + new.amount > totalAmount) then

exception amountError;

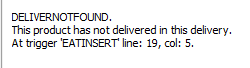
else exit;

end

exception deliverNotFound;

end;

Попытка добавить продукт, которого нет в данной поставке:



Количество потребленного продукта превышает поставленное количество:



**4.**  В соответствии с индивидуальным заданием был создан триггер, который при добавлении продукта в поставку проверяет: нет ли уже этого продукта в поставке. Если есть -вызывается хранимая процедура, которая увеличивает количество уже существующего на величину добавляемого.

drop procedure updlink;

create procedure updLink(link int, am numeric(9,0))

as begin

update linkprod set linkprod.amount = linkprod.amount + :am where linkprod.linkid = :link;

end;

drop trigger productInsert;

create trigger productInsert for linkprod after insert

as

declare variable link int;

begin

for select linkprod.linkid from linkprod

where linkprod.deliver\_id = new.deliver\_id and

linkprod.prodid = new.prodid and

linkprod.linkid != new.linkid

into :link do begin

if (link is null) then exit;

else begin

execute procedure updLink(link, new.amount);

delete from linkprod where linkprod.linkid = new.linkid;

exit;

end

end

end;

Пример выполнения триггера: была попытка добавить продукт 36 в поставку 14 в количестве 1. Запись не была добавлена, вместо этого увеличилось количество данного продукта в предыдущей записи.



**Вывод**

В данной работе были изучены триггеры. В соответствии с программой работы и индивидуальным заданием было создано несколько триггеров.

Триггер — это хранимая процедура особого типа, которую пользователь не вызывает непосредственно, а исполнение которой обусловлено действием по модификации данных: добавлением , удалением строки в заданной таблице, или изменением данных в определенном столбце заданной таблицы реляционной базы данных. Триггеры позволяют контролировать и изменять операции, проводимые над таблицами БД. Триггеры часто используются для обеспечения логической целостности данных в соответствии с предметной областью и реализовывать сложную бизнес-логику.

Триггер может быть в одном из двух состояний активном или неактивном. Запускаются только активные триггеры. По умолчанию триггеры создаются в активном состоянии. Триггер может выполняться в одной из двух фаз, связанных с запрошенными изменениями состояния данных. Ключевое слово BEFORE означает, что триггер вызывается до наступления соответствующего события (событий, если их указано несколько), AFTER — после наступления события (событий).