Санкт-Петербургский Политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт о лабораторной работе №6**

**Дисциплина**: Базы данных

**Тема**: Триггеры

Выполнил студент гр. 43501/1 Фам Конг Минь.

(подпись)

Руководитель Мяснов А.В.

(подпись)

“ ” 2015 г.

Санкт-Петербург

2015

1. **Цель**

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

1. **Программа работы**
2. 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
3. Создать триггер в соответствии с **индивидуальным заданием**, полученным у преподавателя
4. Создать триггер в соответствии с **индивидуальным заданием**, вызывающий хранимую процедуру
5. Выложить скрипт с созданными сущностями в svn
6. Продемонстрировать результаты преподавателю
7. **Выполнение работы**

Вначале была создана тестовая таблица *CAREER* с полями *ID* и NAME типа integer. Далее был написан скрипт для создания триггера для автоматического заполнения поля *ID* в таблице *CAREER.*

set term ^;

CREATE OR ALTER trigger AUTO\_CAREER\_ID for CAREER active before insert AS

begin

select max(CAREER.ID) from CAREER into NEW.ID;

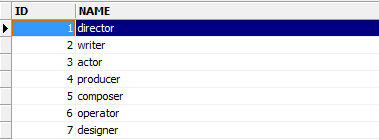
NEW.ID = NEW.ID + 1;

end^

set term ;^

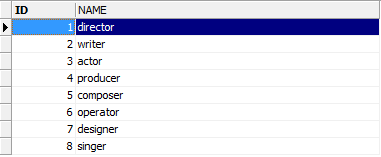
Работа триггера:

SQL> SELECT \* FROM CAREER;



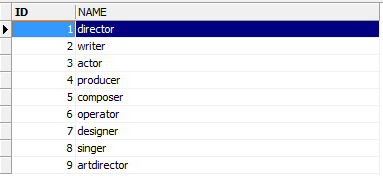
SQL> insert into career values (100, 'singer');

SQL> select \* from CAREER;



SQL> insert into career values (1, 'artdirector');

SQL> select \* from CAREER;



По результатам работы триггера видно, что при добавлении значения в таблицу сначала проверяется последнее (максимальное) значение поля ID, оно инкрементируется, а затем подставляется в добавляемую величину.

Затем был создан триггер для контроля целостности данных в подчиненной таблице при удалении/изменении записей в главной таблице *FILM;*

set term ^;

create or alter exception DEL\_UPD\_FAILED ‘FAILED TO CREATE OR UPDATE THE INFORMATION’^

CREATE OR ALTER trigger CONTROLER for FILM

active before delete or update

AS

begin

if (FILM.F\_ID in (select FILM\_ID from PART)) then

exception DEL\_UPD\_FAILED;

end^

set term ;^

**Индивидуальное задание:**

Реализовать триггеры:

1. При выдаче награды проверять соответствие года фильма и года награды. В случае несоответствия - не добавлять.
2. Пересчитывать рейтинг фильмов по кассовым сборам.

**Первый триггер:**

create exception UPDATEFAILED 'NOT HAVE FILM CREATED IN THIS YEAR';

create trigger NOTUPDATE before insert on award

as

begin

if(new.year\_aw in (select film.year\_of\_creation from

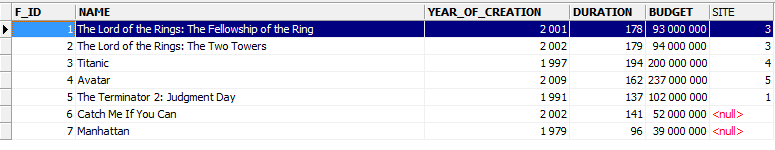
film,award, premium

where award.award\_id = new.award\_id)) then

exception UPDATEFAILED;

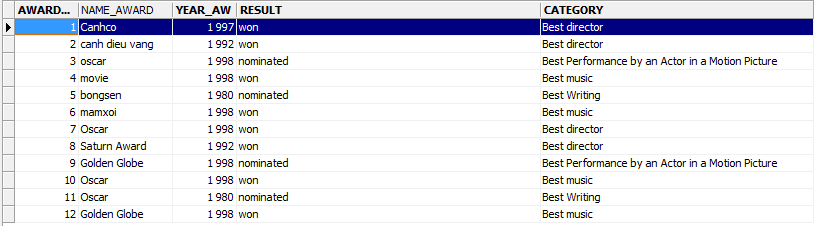
end

данная таблица FILM:



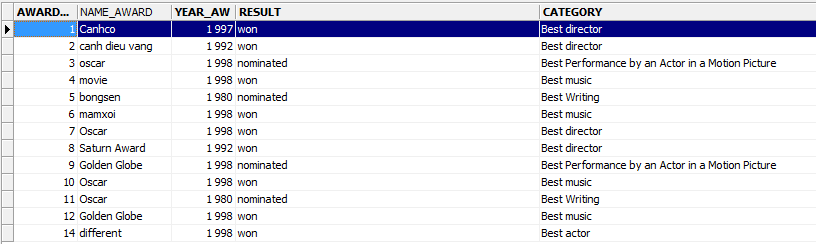
Работа скрипта:

SQL> SELECT \* FROM AWARD ;

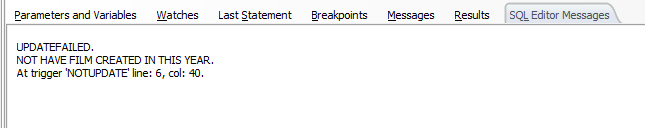


SQL> INSERT INTO award VALUES(14,'different',1998,'won','Best actor');

SQL> SELECT \* FROM AWARD ;



SQL> INSERT INTO award VALUES(15,'different',2001,'won','Best actor');



При попытке добавления в таблицу AWARD новые значения AWARD\_ID, у которых есть соответствие года фильма и года награды данные AWARD\_ID были уже добавлены в таблице AWARD, а на оборот не добавлены.

**Второй триггер:**

SET TERM ^ ;

CREATE OR ALTER TRIGGER FILM\_RATING FOR FILM

ACTIVE AFTER INSERT POSITION 0

as

begin

execute procedure film\_ratings;

end

^

SET TERM ; ^

create or alter procedure FILM\_RATINGS

as

declare variable n int;

declare variable i int;

begin

i=1;

select count(FILM.F\_ID) FROM FILM into :n;

while(i<=n) do

begin

--select film.f\_id from film where f\_id= :i;

update film set film.rating\_film = 0

where film.f\_id=:i ;

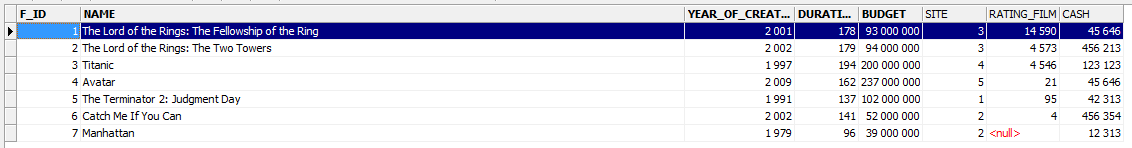
i=i+1;

end

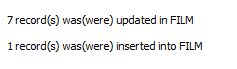
end

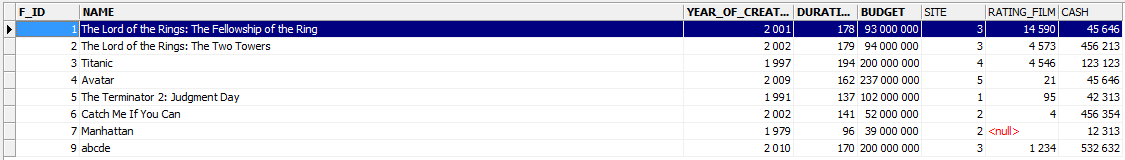
результат выполения:

SELECT \* FROM FILM;



insert into film values(9,'abcde',2010,170,200000000,3,1234,532632) ;





1. **Вывод**

В данной работе мы познакомились с реализацией триггеров. Было создано несколько стандартных триггеров, а так же реализованы триггеры в соответствие с индивидуальным заданием.

С помощью триггеров можно накладывать ограничения на вносимые данные согласно требованиям предметной области БД.

Триггеры полезно использовать для проверки корректности вносимых в БД данных и их целостности.

Также триггеры удобно использовать для оповещения об изменении данных в таблицах.