

Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт о лабораторной работе**

**Дисциплина:** Базы данных

**Тема:** Триггеры, вызовы процедур

Выполнил студент гр. 43501/1

Руководитель

Дроздовский А.А.

Мяснов А.В.

Санкт –Петербург

2016

## 1. Цель работы

Познакомить студентов 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. Ход работы

### 3.1 Триггер для автоматического заполнения ключевого поля

Текст скрипта:

```
create generator increment^
create or alter trigger gener for country
before insert
as
begin
    new.id_country = gen_id(increment,1);
end^
```

```
SQL> set generator increment to 11;
SQL> insert into country values(0,'Algeria');
SQL> select * from country;
```

ID_COUNTRY	COUNTRY
1	Russia
2	Italy
3	USA
4	France
5	Latvia
6	Ukraine
7	Germany
8	China
9	Canada
10	Brazil
11	Abkhazia
12	Algeria

```
SQL> insert into country values(0,'Albania');
SQL> select * from country;
```

ID_COUNTRY	COUNTRY
1	Russia
2	Italy
3	USA
4	France
5	Latvia
6	Ukraine
7	Germany
8	China
9	Canada
10	Brazil
11	Abkhazia
12	Algeria
13	Albania

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

Текст скрипта:

```
create exception error_stage 'ERROR: CANNOT DELETE STAGE TYPE'^
create or alter trigger check_stage for country before delete
as
begin
  if (old.id_country in (select id_country from country_film)) then
    exception error_stage;
  end^
```

```
SQL> delete from country where id_country=3;
Statement failed, SQLSTATE = HY000
exception 1
-ERROR_STAGE
-ERROR: CANNOT DELETE STAGE TYPE
-At trigger 'CHECK_STAGE' line: 4, col: 63
```

## 3.3 Индивидуальное задание

1) При регистрации участия человека в фильме проверять дубли: человек не может участвовать в одном фильме в любых ролях кроме актера более одного раза. Если ограничение не выполняется - выбрасывать исключение

```
create exception error_ind1 'Error: re-entry in the table'^
create or alter trigger Repeat_test for Film_Worker
before insert
as
begin
  If (new.ID_film in (select ID_film from Film_Worker where id_worker=new.id_worker) AND
  new.id_prof=1) then
    exception error_ind1;
  end^
```

```
SQL> insert into Film_Worker values (16,5,1,1500,1);
SQL> insert into Film_Worker values (17,5,2,1333,2);
Statement failed, SQLSTATE = HY000
exception 2
-ERROR_IND1
-Error: re-entry in the table
-At trigger 'REPEAT_TEST' line: 5, col: 103
SQL> select * from Film_worker;
```

ID_FP	ID_FILM	ID_WORKER	SALARY	ID_PROF
1	5	1	1500	1
2	5	2	2000	1
3	1	3	4300	1
4	5	1	5790	2
5	5	2	1333	2
6	5	3	890	2
7	1	4	1290	2
8	1	5	1780	2
9	6	9	7800	1
10	6	6	7800	1
11	6	7	7800	1
12	6	8	7800	1
13	6	10	7800	1
14	6	1	7800	1
15	6	1	7800	1
16	5	1	1500	1

2) При превышении сборов за фильм заданного порогового создавать сиквел с тем же режиссером, 5 актерами с наибольшими гонорарами и остальными создателями фильмов

Процедура *Five\_actor* имеет входные параметры *id\_f* – id фильма, по которому произошли изменения в таблице сборов и *id\_prof* – id профессии, данные по которой требуется узнать. На выходе процедура дает нам 5 работников, которые участвовали в заданном фильме, отсортированные по гонорару.

Процедура *Sum\_collection* подсчитывает общие сборы фильма по всем регионам, по которому были произведены изменения в таблице сборов, затем сравнивает это значение с заданным и в случае, если сборы превысили денежный порог – создает вторую часть данного фильма. Далее загружает 5 самых высокооплачиваемых работников в каждой профессии для данного фильма и заносит их в соответствующую таблицу.

```
--Процедура выведения 5 работников по фильму
CREATE OR ALTER PROCEDURE Five_actor(id_f int, id_prof int)
returns (id_actor int)
as
begin
for
    select first 5 film_worker.id_worker from film_worker
    where film_worker.id_film=:id_f AND film_worker.id_prof=:id_prof
    order by salary desc into :id_actor
do begin
    suspend;
end
end^

--Сравнение сборов с заданным и добавление в таблицы
CREATE OR ALTER PROCEDURE sum_Collections(id_f int, cheak_col int )
as
declare variable sum_col int;
declare variable name_film_next varchar(30);
declare variable id_worker_next int;
declare variable id_next int;
declare variable id_next_worker int;
begin
    select Name from Films where id=:id_f into :name_film_next;
    name_film_next:=name_film_next || ' 2';
    select max(id) from FILMS into :id_next;
    id_next=:id_next+1;
    sum_col=0;
    select sum(collections) from Collections where id_film=:id_f AND id_period=4
    into :sum_col;
    if(:cheak_col<:sum_col AND :name_film_next not in (select Name from Films )) then
    begin
        insert into Films values (:id_next,:name_film_next,null,' ',null,null,null);
        for select * from Five_actor(:id_f,1) into :id_worker_next
        do begin
            select max(id_fp) from Film_Worker into :id_next_worker;
            id_next_worker=:id_next_worker+1;
            insert into Film_Worker values (:id_next_worker,:id_next,:id_worker_next,null,1);
        end
        for select * from Five_actor(:id_f,2) into :id_worker_next
        do begin
            select max(id_fp) from Film_Worker into :id_next_worker;
            id_next_worker=:id_next_worker+1;
            insert into Film_Worker values (:id_next_worker,:id_next,:id_worker_next,null,2);
        end
    end
end
end^
```

```
--Триггер срабатывающий при добавление в таблицу Сборов нового значения
create or alter trigger add_film for Collections after insert
as
begin
execute procedure sum_Collections(new.id_film,1000000);
end^
```

Результат работы:

Таблицы до изменений:

```
SQL> select * from films;
-----
```

ID	NAME	YEARS	TAGLINE	AGE_LIMIT	TIMES	BUDGET
1	The Shawshank Redemption	1994	Fear - it shackles. Hope - is freedom	16	142	2500000
2	The Green Mile	1999	Paul Edgecombe did not believe in miracles. While not faced with one of them.	16	189	60000000
3	Forrest Gump	1994	The world will never be the same after you see it through the eyes of Forrest Gump	12	142	55000000
4	Schindlers List	1993	This list of - life	16	195	22000000
5	Intouchables	2011	Sometimes you have to reach into someone else is world to find out what is missing in your own	16	112	9500000
6	Assassins Creed	2016	Welcome to the Spanish Inquisition	16	115	125000000
7	Passengers	2016	Nothing is by chance	16	116	110000000
8	CCC	2011	aa	16	112	9500000
9	DDD	2011	aa	16	112	9500000
10	EEE	2011	aa	16	112	9500000
11	FFF	2011	aa	16	112	9500000
12	BBB	2012	aa	12	111	9500000

```
SQL> select * from film_worker;
-----
```

ID_FP	ID_FILM	ID_WORKER	SALARY	ID_PROF
1	5	1	1500	1
2	5	2	2000	1
3	1	3	4300	1
4	5	1	5700	2
5	5	2	1333	2
6	5	3	890	2
7	1	4	1290	2
8	1	5	1780	2
9	6	9	7800	1
10	6	6	7800	1
11	6	7	7800	1
12	6	8	7800	1
13	6	10	7800	1
14	6	1	7800	1
15	6	1	7800	1

Добавим информацию о сборах для фильма «Intouchables»:

```
SQL> insert into Collections values(13,8,4,289000,5);
SQL> select * from films;
-----
```

ID	NAME	YEARS	TAGLINE	AGE_LIMIT	TIMES	BUDGET
1	The Shawshank Redemption	1994	Fear - it shackles. Hope - is freedom	16	142	2500000
2	The Green Mile	1999	Paul Edgecombe did not believe in miracles. While not faced with one of them.	16	189	60000000
3	Forrest Gump	1994	The world will never be the same after you see it through the eyes of Forrest Gump	12	142	55000000
4	Schindlers List	1993	This list of - life	16	195	22000000
5	Intouchables	2011	Sometimes you have to reach into someone else is world to find out what is missing in your own	16	112	9500000
6	Assassins Creed	2016	Welcome to the Spanish Inquisition	16	115	125000000
7	Passengers	2016	Nothing is by chance	16	116	110000000
8	CCC	2011	aa	16	112	9500000
9	DDD	2011	aa	16	112	9500000
10	EEE	2011	aa	16	112	9500000
11	FFF	2011	aa	16	112	9500000
12	BBB	2012	aa	12	111	9500000

Как видно выше, не произошло никаких изменений, т.к. сборы данного фильма не превысили 1000000. Проверим это и добавим еще информацию по сборам:

```
SQL> select sum(collections) from Collections where id_film=5 AND id_period=4;

SUM
=====
289000
```

```
SQL> insert into Collections values(14,5,4,345000,5);
SQL> insert into Collections values(15,6,4,178000,5);
SQL> select sum(collections) from Collections where id_film=5 AND id_period=4;

SUM
=====
812000
```

ID	NAME	YEARS	TAGLINE
1	The Shawshank Redemption	1994	Fear - it shackle
2	The Green Mile	1999	Paul Edgecombe di
3	Forrest Gump	1994	The world will ne
4	Schindlers List	1993	This list of - li
5	Intouchables	2011	Sometimes you hav
6	Assassins Creed	2016	Welcome to the Sp
7	Passengers	2016	Nothing is by cha
8	CCC	2011	aa
9	DDD	2011	aa
10	EEE	2011	aa
11	FFF	2011	aa
12	BBB	2012	aa

Как видно общие сборы фильма выросли до 812000, но этого по-прежнему не хватает для создания второй части фильма, добавим еще информацию по сборам, чтобы перейти порог в 1000000:

```
SQL> insert into Collections values(16,7,4,188001,5);
SQL> select * from films;
```

ID	NAME	YEARS	TAGLINE	AGE_LIMIT	TIMES	BUDGET
1	The Shawshank Redemption	1994	Fear - it shackles. Hope - is freedom	16	142	2500000
2	The Green Mile	1999	Paul Edgecombe did not believe in miracles. While not faced with one of them.	16	189	60000000
3	Forrest Gump	1994	The world will never be the same after you see it through the eyes of Forrest Gump	12	142	55000000
4	Schindlers List	1993	This list of - life	16	195	22000000
5	Intouchables	2011	Sometimes you have to reach into someone else is world to find out what is missing in your own	16	112	9500000
6	Assassins Creed	2016	Welcome to the Spanish Inquisition	16	115	125000000
7	Passengers	2016	Nothing is by chance	16	116	110000000
8	CCC	2011	aa	16	112	9500000
9	DDD	2011	aa	16	112	9500000
10	EEE	2011	aa	16	112	9500000
11	FFF	2011	aa	16	112	9500000
12	BBB	2012	aa	12	111	9500000
13	Intouchables 2	<null>	<null>	<null>	<null>	<null>

```
SQL> select * from film_worker;
```

ID_FP	ID_FILM	ID_WORKER	SALARY	ID_PROF
1	5	4	1500	2
2	5	5	2000	2
3	1	3	4300	1
4	5	1	5790	1
5	5	2	1333	1
6	5	3	890	1
7	1	4	1290	2
8	1	5	1780	2
9	6	9	7800	1
10	6	6	7800	1
11	6	7	7800	1
12	6	8	7800	1
13	6	10	7800	1
14	6	1	7800	1
16	5	11	4390	2
17	5	12	3200	2
18	13	1	<null>	1
19	13	2	<null>	1
20	13	3	<null>	1
21	13	11	<null>	2
22	13	12	<null>	2
23	13	5	<null>	2
24	13	4	<null>	2

Проверим информацию по сборам:

```
SQL> select sum(collections) from Collections where id_film=5 AND id_period=4;
```

```

SUM
=====
1000001

```

При дальнейшем добавление сборов для данного фильма копии второй части создаваться не будут:

```
SQL> insert into Collections values(17,1,4,25000,5);
SQL> select * from films;
```

ID	NAME	YEARS	TAGLINE
1	The Shawshank Redemption	1994	Fear - i
2	The Green Mile	1999	Paul Edg
3	Forrest Gump	1994	The worl
4	Schindlers List	1993	This lis
5	Intouchables	2011	Sometime
6	Assassins Creed	2016	Welcome
7	Passengers	2016	Nothing
8	CCC	2011	aa
9	DDD	2011	aa
10	EEE	2011	aa
11	FFF	2011	aa
12	BBB	2012	aa
13	Intouchables 2		<null>

```
SQL> select * from film_worker;
```

ID_FP	ID_FILM	ID_WORKER	SALARY	ID_PROF
1	5	4	1500	2
2	5	5	2000	2
3	1	3	4300	1
4	5	1	5790	1
5	5	2	1333	1
6	5	3	890	1
7	1	4	1290	2
8	1	5	1780	2
9	6	9	7800	1
10	6	6	7800	1
11	6	7	7800	1
12	6	8	7800	1
13	6	10	7800	1
14	6	1	7800	1
16	5	11	4390	2
17	5	12	3200	2
18	13	1	<null>	1
19	13	2	<null>	1
20	13	3	<null>	1
21	13	11	<null>	2
22	13	12	<null>	2
23	13	5	<null>	2
24	13	4	<null>	2

Созданные триггеры в ходе работы:

```
SQL> show triggers;
```

Table name	Trigger name	Invalid
COLLECTIONS	ADD_FILM	
COUNTRY	CHECK_STAGE	
COUNTRY	GENER	
FILM_WORKER	REPEAT_TEST	

#### **4. Выводы**

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