Санкт-Петербургский политехнический университет Петра Великого

Институт информационных технологий и управления

Кафедра компьютерных систем и программных технологий

**Отчёт лабораторной работе №5**

**Дисциплина**: Базы данных

**Тема**: Триггеры, вызовы процедур

Выполнил студент гр. 43501/1 С.И.Бочкова

(подпись)

Руководитель А.В. Мяснов

(подпись)

“ ” 2015 г.

Санкт - Петербург

2015

1. **Цель работы**

Изучить возможности триггеров, выполнить полученное задание

1. **Проведение работы**

*Создать триггер для автоматического заполнения ключевого поля*

SET SQL DIALECT 3;

CREATE GENERATOR GEN\_FILM\_ID;

SET TERM ^ ;

CREATE OR ALTER TRIGGER NEW\_FILM FOR FILM

ACTIVE BEFORE INSERT POSITION 0

AS

begin

IF (NEW.f\_id IS NULL) THEN

NEW.f\_id = GEN\_ID(GEN\_FILM\_ID,1);

end

^

SET TERM ; ^

Триггер для контроля целостности данных в подчиненной таблице при удалении записей в главной таблице (таблице фильмов):

SET SQL DIALECT 3;

SET TERM ^ ;

CREATE OR ALTER TRIGGER DELETE\_FILM FOR FILM

ACTIVE AFTER DELETE POSITION 0

AS

BEGIN

DELETE FROM soundtrack WHERE film = OLD.f\_id;

DELETE FROM PRODUCTION WHERE FILM\_ID = OLD.f\_id;

DELETE FROM premium WHERE FILM\_ID = OLD.f\_id;

DELETE FROM part WHERE FILM\_ID = OLD.f\_id;

DELETE FROM made\_in WHERE FILM\_ID = OLD.f\_id;

DELETE FROM style WHERE FILM\_ID = OLD.f\_id;

END

^

SET TERM ; ^

Триггер для контроля целостности данных в подчиненной таблице при изменении записей в главной таблице (таблице фильмов):

SET SQL DIALECT 3;

SET TERM ^ ;

CREATE OR ALTER TRIGGER UPDATE\_FILM FOR FILM

ACTIVE AFTER UPDATE POSITION 0

AS

begin

IF (OLD.f\_id <> NEW.f\_id) THEN

begin

UPDATE soundtrack SET film = NEW.f\_id WHERE film = OLD.F\_ID;

UPDATE style SET FILM\_ID = NEW.f\_id WHERE FILM\_ID = OLD.F\_ID;

UPDATE production SET FILM\_ID = NEW.f\_id WHERE FILM\_ID = OLD.F\_ID;

UPDATE premium SET FILM\_ID = NEW.f\_id WHERE FILM\_ID = OLD.F\_ID;

UPDATE part SET FILM\_ID = NEW.f\_id WHERE FILM\_ID = OLD.F\_ID;

UPDATE made\_in SET FILM\_ID = NEW.f\_id WHERE FILM\_ID = OLD.F\_ID;

end

end

^

SET TERM ; ^

Примеры работы триггеров:

1)INSERT INTO film (name,year\_of\_creation,duration,budget)

VALUES ('Manhattan',1979,96,39000000);

В результате работы триггера в таблицу фильмов добавится ещё один фильм с уникальным ключевым полем.

2) DELETE FROM soundtrack WHERE name LIKE '%My%';

В результате работы триггера из всех подчиненных таблиц удалятся записи, содержащие внешние ключи на удаляемый фильм.

3) UPDATE soundtrack SET sound\_id=5 WHERE sound\_id = 0;

При выполнении модификации записей в главной таблице, в подчиненных таблицах изменяются записи, содержащие внешние ключи на изменяемый саундтрек.

Индивидуальное задание:

1. Автоматическое заполнение первичного ключа при добавлении записей для одной из таблиц.

SET SQL DIALECT 3;

CREATE GENERATOR GEN\_FILM\_ID;

SET TERM ^ ;

CREATE OR ALTER TRIGGER NEW\_FILM FOR FILM

ACTIVE BEFORE INSERT POSITION 0

AS

begin

IF (NEW.f\_id IS NULL) THEN

NEW.f\_id = GEN\_ID(GEN\_FILM\_ID,1);

end

^

SET TERM ; ^

1. Создать отдельный жанр "10 наиболее кассовых". При обновлении данных по прокатным сборам фильмов добавлять указанный жанр к фильму, если он входит в десять наиболее кассовых. Удалять указанный жанр у фильма, который выпал из десяти наиболее кассовых.

**3.Вывод**

Триггеры – это важная часть любой СУБД, позволяющая:

- реализовать сложные ограничения целостности данных, которые невозможно осуществить через описательные ограничения, устанавливаемые при создании таблиц (например, нецикличность маршрута);

- уменьшить код транзакций, т.к., например, при удалении заказа из таблицы всегда необходимо каскадно удалять и соответствующие билеты. Код для удаления соответствующих путей всегда один и тот же. Ко всему прочему этот код уже будет скомпилирован, что ускорит выполнение транзакции ;

- упростить заполнение БД, проводя расчет для тех полей, значение которых может быть рассчитано алгоритмически (например, расчет цены);

- отслеживать модификации таблиц или выполнять действия не связанные с БД.