Санкт-Петербургский политехнический университет Петра Великого

Кафедра компьютерных систем и программных технологий

**Отчёт по лабораторной работе №6**

**Дисциплина**: Базы данных

**SQL-программирование: Триггеры, вызовы процедур**

Выполнил студент гр. 43501/3 Анисимов А.А.

Преподаватель: Мяснов А. В.

Санкт-Петербург

2016

**Цели 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

**Индивидуальное задание:**

Реализовать следующие триггеры:

1. Реализовать проверку наличия доступа при внесении записей в историю.
2. При создании договора создавать автоматически группу для владельца и устанавливать тип доступа как индивидуальный.

**Выполнение 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

* Триггер для автоматического заполнения поля ITEM\_ID таблицы(ITEM).

**Листинг 1. Триггер заполнения ITEM\_ID**

|  |
| --- |
| CREATE SEQUENCE create\_id;  SET GENERATOR create\_id TO 100010;  CREATE TRIGGER auto\_create\_id FOR item BEFORE INSERT  AS  BEGIN  NEW.item\_id = GEN\_ID(create\_id, 1);  END |

Проверим работу триггера:

Воспользуемся средствами IBExpert для добавления нового предмета:



Как видно поле ITEM\_ID оставлено со значением NULL.

Сделаем коммит.

Полученные результаты:



Поле contract\_id получило значение 100011. Триггер работает.

* Триггер, который проверяет данные на целостность: при попытке удаления не пустой депозитной ячейки срабатывает триге:

**Листинг 2. Триггер контроля**

|  |
| --- |
| CREATE EXCEPTION error\_depb\_non\_empty 'Error. This deposit box does not empty!';  CREATE TRIGGER control\_depb\_emptiness FOR contract BEFORE DELETE  AS  BEGIN  AS  begin  /\* Trigger text \*/  if(OLD.DEPB\_VALUE > 0.0) then  exception error\_depb\_non\_empty;  END |

Проверка работы триггера:

Таблица items:

|  |
| --- |
| SQL> delete from deposit\_box where depb\_id = 2;  Statement failed, SQLSTATE = HY000  exception 1  -ERROR\_DEPB\_NON\_EMPTY  -Error. This deposit box does not empty!  -At trigger 'CONTROL\_DEPB\_EMPTINESS' line: 6, col: 28 |

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

**Реализовать проверку наличия доступа при внесении записей в историю.**

**Листинг 3.**

|  |
| --- |
| SET TERM ^ ;  CREATE OR ALTER TRIGGER ACCESS\_CHECK FOR BOX\_HISTORY  ACTIVE BEFORE INSERT POSITION 0  AS  begin  /\* Trigger text \*/  /\* if only\_client client have an access, than > 1 client in access group generate error \*/  IF ((SELECT accesstype\_id FROM contract WHERE NEW.contract\_id = contract.contract\_id) = 1) THEN  IF ((SELECT COUNT(\*) FROM CLIENTS\_IN\_GROUP\_OF\_ACCESS(NEW.access\_group\_id)) > 1) THEN  EXCEPTION error\_more\_than\_one\_client;  /\* if group\_only, we need > 2 clients \*/  IF ((SELECT accesstype\_id FROM contract WHERE NEW.contract\_id = contract.contract\_id) = 2) THEN  IF ((SELECT COUNT(\*) FROM CLIENTS\_IN\_GROUP\_OF\_ACCESS(NEW.access\_group\_id)) < 2) THEN  EXCEPTION error\_not\_enough\_clients;  /\* if client\_and\_other\_client, we need 2 clients \*/  IF ((SELECT accesstype\_id FROM contract WHERE NEW.contract\_id = contract.contract\_id) = 3) THEN  IF ((SELECT COUNT(\*) FROM CLIENTS\_IN\_GROUP\_OF\_ACCESS(NEW.access\_group\_id)) <> 2) THEN  EXCEPTION error\_two\_clients\_req;  /\* if client\_and\_other\_group \*/  IF ((SELECT accesstype\_id FROM contract WHERE NEW.contract\_id = contract.contract\_id) = 4) THEN  IF ((SELECT COUNT(\*) FROM CLIENTS\_IN\_GROUP\_OF\_ACCESS(NEW.access\_group\_id)) < 2) THEN  EXCEPTION error\_client\_and\_group\_req;  /\* if list\_of\_clients \*/  IF ((SELECT accesstype\_id FROM contract WHERE NEW.contract\_id = contract.contract\_id) = 5) THEN  IF ((SELECT COUNT(\*) FROM CLIENTS\_IN\_GROUP\_OF\_ACCESS(NEW.access\_group\_id)) < 1) THEN  EXCEPTION error\_list\_of\_clients\_req;  /\* if with\_specific\_client\_only \*/  IF ((SELECT accesstype\_id FROM contract WHERE NEW.contract\_id = contract.contract\_id) = 6) THEN  IF ((SELECT COUNT(\*) FROM CLIENTS\_IN\_GROUP\_OF\_ACCESS(NEW.access\_group\_id)) <> 2) THEN  EXCEPTION error\_specific\_client\_req;  /\* \*/  IF ((SELECT accesstype\_id FROM contract WHERE NEW.contract\_id = contract.contract\_id) = 7) THEN  IF ((SELECT COUNT(\*) FROM CLIENTS\_IN\_GROUP\_OF\_ACCESS(NEW.access\_group\_id)) < (SELECT access\_type.access\_min\_num FROM access\_type)) THEN  EXCEPTION error\_min\_amount\_req;  end^  SET TERM ; ^ |

Проверка работы триггера. Попытаемся добавить запись в историю, для контракта с типом доступа требующим минимального значения клиентов 5, при наличии в группе 2х клиентов:

|  |
| --- |
| SQL> INSERT INTO box\_history (history\_id, client\_id, history\_previousValue, history\_currentValue, depb\_id, contract\_id) VALUES (100010, 56800, 56.17, 3333, 10350, 11);  Statement failed, SQLSTATE = HY000  exception 9  -ERROR\_MIN\_AMOUNT\_REQ  -Error! Not enough clients for access!  -At trigger 'ACCESS\_CHECK' line: 36, col: 1 |

**При создании договора создавать автоматически группу для владельца и устанавливать тип доступа как индивидуальный.**

**Листинг 4.**

|  |
| --- |
| SET TERM ^ ;  CREATE OR ALTER TRIGGER AUTO\_GROUP\_AND\_ACCTYPE FOR CONTRACT  ACTIVE BEFORE INSERT POSITION 0  AS  declare variable gr\_id integer;  begin  /\* Trigger text \*/  gr\_id = GEN\_ID(group\_id, 1);  NEW.accesstype\_id = 1;  INSERT INTO groups (group\_id) VALUES (:gr\_id);  INSERT INTO group\_member (gm\_id, group\_id, client\_id) VALUES (GEN\_ID(gm\_id, 1), :gr\_id, NEW.client\_id);  NEW.group\_id = :gr\_id;  end^  SET TERM ; ^ |

Проверка работы триггера:

До коммита:



После:



**Выводы:**

В данной работе были изучены триггеры и генераторы.

Триггер – это хранимая процедура особого типа, которую пользователь не вызывает непосредственно, а исполнение которой обусловлено действием по модификации данных.

Это мощный инструмент для сохранения целостности и повышения безопасности базы данных.