

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

_____ Данг Хань
(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.

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

3. Выполнение работы

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

```
set term ^;
CREATE OR ALTER trigger AUTOINCR for AUTOTRIG active before insert AS
begin
    select max(AUTOTRIG.INCR) from AUTOTRIG into NEW.INCR;
    NEW.INCR = NEW.INCR + 1;
end^
set term ;^
```

Работа триггера:

```
SQL> select * from AUTOTRIG;

          INCR          ZNACH
=====
          1             20

SQL> insert into AUTOTRIG values (100, 50);
SQL> select * from AUTOTRIG;

          INCR          ZNACH
=====
          1             20
          2             50

SQL> insert into AUTOTRIG values (1, 45);
SQL> select * from AUTOTRIG;
```

INCR	ZNACH
=====	=====
1	20
2	50
3	45

По результатам работы триггера видно, что при добавлении значения в таблицу сначала проверяется последнее (максимальное) значение поля *INCR*, 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 *MODELS*:

```
set term ^;
create or alter exception DEL_UPD_FAILED 'FAILED TO CREATE OR UPDATE
THE INFORMATION' ^
CREATE OR ALTER trigger CONTRZEL for MODELS
active before delete or update
AS
begin
    if (OLD.ID in (select Models_Code from CARS)) then
        exception DEL_UPD_FAILED;
end^
set term ;^
```

Индивидуальное задание:

Реализовать триггеры:

1. При добавлении книги в заказ проверять есть ли такая книга уже в заказе. Если есть - увеличивать количество этой книги в заказе, а повторно одну и ту же книгу не добавлять.
2. Обновлять рейтинг автора с учетом продаж его книг за последний год.

Первый триггер:

```
connect 'D:/bs.fdb' user 'SYSDBA' password 'masterkey';
create or alter trigger lab61 before insert on list_cart
as
begin
    if (new.id_book in (select id_book from list_cart)) then
        update list_cart set
        list_cart.sum_order=new.sum_order+list_cart.sum_order where
        list_cart.id_book=new.id_book;
end
```

```

connect 'D:/bs.fdb' user 'SYSDBA' password 'masterkey';
CREATE trigger lab62 after insert on list_cart
as
declare variable idd varchar(10);
declare variable cc integer;
begin
    idd= new.id_book;
    select count(*) from list_cart where list_cart.id_book =:idd into
:cc;
    if (:cc >1) then
        DELETE FROM list_cart WHERE
list_cart.id_list_cart=new.id_list_cart;
End

```

Результат выполнения

Данные в таблице заказе до добавления

Table : [LIST_CART] : BOOK STORE (localhost:D:\BS.FDB)

Table | Get record count | LIST_CART

Fields | Constraints | Indices | Dependencies | Triggers | Data | Master/Detail View

Record: 1

ID_LIST_CART	ID_BOOK	ID_CART	SUM_ORDER
LC001	B001	CA005	1 000
LC002	B002	CA002	800
LC003	B004	CA003	900
LC004	B003	CA003	600
LC005	B005	CA002	400
LC006	B013	CA004	750
LC007	B009	CA007	850
LC008	B012	CA010	860
LC009	B008	CA001	550
LC010	B006	CA008	350
LC011	B007	CA004	630
LC012	B010	CA010	570

Grid View | Form View | Print Data

Выполнил команды добавить

```
Script Statements Options
connect 'D:/bs.fdb' user 'SYSDBA' password 'masterkey';
insert into list cart values ('LC013','B001','CA005',2);
```

И результат получил

Table : [LIST_CART] : BOOK STORE (localhost:D:\BS.FDB)

Table Constraints Indices Dependencies Triggers Data Master/Detail View

Record: 1

ID_LIST_CART	ID_BOOK	ID_CART	SUM_ORDER
LC001	B001	CA005	1 002
LC002	B002	CA002	800
LC003	B004	CA003	900
LC004	B003	CA003	600
LC005	B005	CA002	400
LC006	B013	CA004	750
LC007	B009	CA007	850
LC008	B012	CA010	860
LC009	B008	CA001	550
LC010	B006	CA008	350
LC011	B007	CA004	630
LC012	B010	CA010	570

Grid View Form View Print Data

При добавлении другое значение

```
Script Statements Options
connect 'D:/bs.fdb' user 'SYSDBA' password 'masterkey';
insert into list cart values ('LC013','B011','CA005',2);
```

И реультат получил

Table : [LIST_CART] : BOOK STORE (localhost:D:\BS.FDB)

Table | Get record count LIST_CART

Fields | Constraints | Indices | Dependencies | Triggers | Data | Master/Detail View

Record: 1

ID_LIST_CART	ID_BOOK	ID_CART	SUM_ORDER
LC001	B001	CA005	1 002
LC002	B002	CA002	800
LC003	B004	CA003	900
LC004	B003	CA003	600
LC005	B005	CA002	400
LC006	B013	CA004	750
LC007	B009	CA007	850
LC008	B012	CA010	860
LC009	B008	CA001	550
LC010	B006	CA008	350
LC011	B007	CA004	630
LC012	B010	CA010	570
LC013	B011	CA005	2

Grid View | Form View | Print Data

Второй процедур:

Текты процедура

Обновлять рейтинг автора с учетом продаж его книг за последний год

```
connect 'D:/bs.fdb' user 'SYSDBA' password 'masterkey';
create procedure order_rating_new (id_au varchar(10))
returns( SUUM INTEGER)
as
declare variable Maxx_year integer;
declare variable nam date;
declare variable sum_zakat integer;
begin
sum_zakat=0;
select max (date_dostavki) from cart into :nam;
Maxx_year= extract (year from :nam);
select sum(sum_order) from list_cart, cart, list_authors where
(list_cart.id_book=list_authors.id_book)
and (list_authors.id_of_author=:id_au) and (list_cart.id_cart=cart.id_cart)
and (extract (year from(cart.date_dostavki))=:Maxx_year) into :sum_zakat;
if (:sum_zakat!=0) then suum=:sum_zakat;
else suum=0;
```

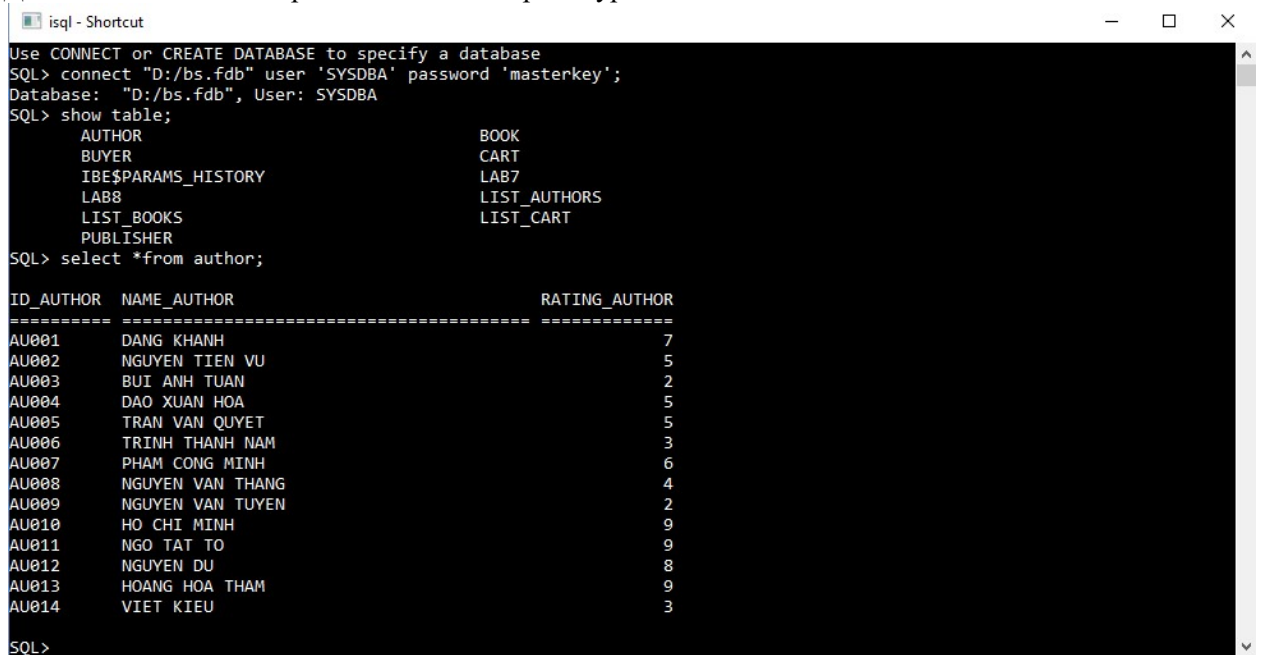
```

if (suum>1000) then update author set author.rating_author=10 where
author.id_author=:id_au;
else if (suum>900) then update author set author.rating_author=9 where
author.id_author=:id_au;
else if (suum>800) then update author set author.rating_author=8 where
author.id_author=:id_au;
else if (suum>700) then update author set author.rating_author=7 where
author.id_author=:id_au;
else if (suum>600) then update author set author.rating_author=6 where
author.id_author=:id_au;
else if (suum<600) then update author set author.rating_author=5 where
author.id_author=:id_au;
suspend;
end

```

Пример результат выполнения

Данные в таблице автора до вызывания процедур



```

isql - Shortcut
Use CONNECT or CREATE DATABASE to specify a database
SQL> connect "D:/bs.fdb" user 'SYSDBA' password 'masterkey';
Database: "D:/bs.fdb", User: SYSDBA
SQL> show table;
AUTHOR          BOOK
BUYER           CART
IBES$PARAMS_HISTORY LAB7
LAB8            LIST_AUTHORS
LIST_BOOKS      LIST_CART
PUBLISHER
SQL> select *from author;

ID_AUTHOR  NAME_AUTHOR          RATING_AUTHOR
-----
AU001      DANG KHANH           7
AU002      NGUYEN TIEN VU       5
AU003      BUI ANH TUAN         2
AU004      DAO XUAN HOA         5
AU005      TRAN VAN QUYET       5
AU006      TRINH THANH NAM      3
AU007      PHAM CONG MINH       6
AU008      NGUYEN VAN THANG     4
AU009      NGUYEN VAN TUYEN     2
AU010      HO CHI MINH          9
AU011      NGO TAT TO           9
AU012      NGUYEN DU            8
AU013      HOANG HOA THAM       9
AU014      VIET KIEU            3
SQL>

```

И после выполнения процедура, получил результат

```
isql - Shortcut
AU012      NGUYEN DU      8
AU013      HOANG HOA THAM 9
AU014      VIET KIEU      3

SQL> execute procedure order_rating_new('AU001');

      SUUM
=====
      1002

SQL> select *from author;

ID_AUTHOR  NAME_AUTHOR                                     RATING_AUTHOR
=====
AU001      DANG KHANH                                     10
AU002      NGUYEN TIEN VU                               5
AU003      BUI ANH TUAN                                2
AU004      DAO XUAN HOA                                5
AU005      TRAN VAN QUYET                              5
AU006      TRINH THANH NAM                             3
AU007      PHAM CONG MINH                              6
AU008      NGUYEN VAN THANG                             4
AU009      NGUYEN VAN TUYEN                             2
AU010      HO CHI MINH                                 9
AU011      NGO TAT TO                                  9
AU012      NGUYEN DU                                   8
AU013      HOANG HOA THAM                              9
AU014      VIET KIEU                                   3

SQL>
```

Видно что рейтинг автора с id='AU001' изменился.

Или другой пример

```
SQL> execute procedure order_rating_new('AU004');

      SUUM
=====
      802

SQL>
SQL> select *from author;

ID_AUTHOR  NAME_AUTHOR                                     RATING_AUTHOR
=====
AU001      DANG KHANH                                     10
AU002      NGUYEN TIEN VU                               5
AU003      BUI ANH TUAN                                9
AU004      DAO XUAN HOA                                8
AU005      TRAN VAN QUYET                              5
AU006      TRINH THANH NAM                             3
AU007      PHAM CONG MINH                              6
AU008      NGUYEN VAN THANG                             4
AU009      NGUYEN VAN TUYEN                             2
AU010      HO CHI MINH                                 9
AU011      NGO TAT TO                                  9
AU012      NGUYEN DU                                   8
AU013      HOANG HOA THAM                              9
AU014      VIET KIEU                                   3

SQL>
```

Видно что рейтинг автора с id='AU004' изменился.

4. Вывод

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.