Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе№6

По дисциплине «Базы данных»

«Триггеры, вызовы процедур»

Работу выполнили студенты группы №43501/4 Н.С. Шаляпин_____

Работу принял преподаватель А.В. Мяснов_____

Санкт-Петербург

2015

1. Цель работы

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

2 . Триггеры

Триггер (англ, trigger) — это хранимая процедура особого типа, которую пользователь не вызывает непосредственно, а исполнение которой обусловлено действием по модификации данных: добавлением INSERT, удалением DELETEстроки в заданной таблице, или изменением UPDATEданных в определенном столбце заданной таблицы реляционной базы данных. Триггеры применяются для обеспечения целостности данных и реализации сложной бизнес-логики. Триггер запускается сервером автоматически при попытке изменения данных в таблице, с которой он связан. Все производимые им модификации данных рассматриваются как выполняемые в транзакции, в которой выполнено действие, вызвавшее срабатывание триггера. Соответственно, в случае обнаружения ошибки или нарушения целостности данных может произойти откат этой транзакции.

Момент запуска триггера определяется с помощью ключевых слов BEFORE(триггер запускается до выполнения связанного с ним события; например, до добавления записи) или AFTER(после события). В случае, если триггер вызывается до события, он может внести изменения в модифицируемую событием запись (конечно, при условии, что событие — не удаление записи). Некоторые СУБД накладывают ограничения на операторы, которые могут быть использованы в триггере (например, может быть запрещено вносить изменения в таблицу, на которой «висит» триггер, и т, п.).

Кроме того, триггеры могут быть привязаны не к таблице, а к представлению (VIEW). В этом случае с их помощью реализуется механизм «обновляемого представления». В этом случае ключевые слова BEFOREи AFTERвлияют лишь на последовательность вызова триггеров, так как собственно событие (удаление, вставка или обновление) не происходит.

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 Ход работы

- Был создан триггер, который автоматически заполняет поле marks_idтаблицы marks(имитация автоинкремента):

Триггер автоинкремента.

```
CREATE GENERATOR gen_marks_id;
SET GENERATOR gen_marks_id TO 0;
set term !! ;
CREATE TRIGGER marks_bi FOR marks
ACTIVE BEFORE INSERT POSITION 0
AS
BEGIN
NEW.marks_id = GEN_ID(gen_marks_id, 1);
END!!
setterm ; !!
```

- Был создан триггер, который проверяет данные на целостность: при попытке удаления или изменения записи в таблице марок, на которую присутствуют внешние ссылки, он выдает ошибку.

+Триггер проверки целостности.

```
create trigger control_del_mark for marks before delete or update
as
begin
if (OLD.marks_id in ( select model_car.marks_id from model_car )) then
exception exc_err;
end;
```

- В соответствии с индивидуальным заданием был создан триггер, который при проверке дублей при добавлении опции к заказу. При дубле - не добавлять.

+Триггерпроверкинацелостность

```
create trigger checkDubles before insert on trade_additional_options
AS
declare variable id int;
BEGIN
EXECUTE PROCEDURE forTrigCheckDubl(new.additional_options_id)
RETURNING_VALUES :id;
if(id is null) then exit;
else exception exc_err;
END

create PROCEDURE forTrigCheckDubl(additional_options_id INTEGER)
```

```
returns (iddint)
as
BEGIN
select trade_additional_options.trade_additional_options_id from trade_additional_options
where additional_options_id =: additional_options_id
rows 1 into idd;
END
```

- В соответствии с индивидуальным заданием был создан триггер, который при попытке добавить в заказ опцию, которая уже есть в выбранной комплектации не осуществляет добавление.

+Триггерпроверкинацелостность

```
create exception check_opt_ex 'inserting option that already in deal ';
drop trigger check_opt ;
create trigger check_opt before insert on trade_additional_options
as
begin
if (new.additional_options_id not in ( select trade_additional_options.additional_options_id
from trade_additional_options
wheretrade_additional_options.trade_id = new.trade_id)) then
exit ;
exception check_opt_ex ;
end;
```

4. Вывод

В данной работе были изучены триггеры и генераторы.

Триггер — это хранимая процедура особого тина, которую пользователь не вызывает непосредственно, а исполнение которой обусловлено действием но модификации данных. Триггеры позволяют контролировать и изменять операции, проводимые над таблицами БД. Например, триггеры позволяют симулировать автоинкрементирующиеся ключи таблицы (firebirdne поддерживает автоинкрементирование), что очень удобно при добавлении новых записей в таблицу. Так же триггеры позволяют обеспечивать логическую целостность данных в соответствии с предметной областью и реализовывать сложную бизнес-логику.

Генератор — это специальный объект базы данных, который генерирует уникальные последовательные числа. Одним из применений генераторов является их использование в триггерах автоинкрементирования ключей. В таких триггерах необходимо использовать генераторы, так как они обеспечивают уникальность генерируемых значений даже при параллельной обработке нескольких запросов.

При выполнении работы были созданы триггеры в соответствии с индивидуальным заданием. Проблем в ходе выполнения работы не возникло.