Санкт-Петербургский государственный политехнический университет

Кафедра компьютерных систем и программных технологий

**Отчёт лабораторной работе №5**

**Дисциплина**: Базы данных

**Тема**: Триггеры, вызовы процедур

Выполнил студент гр. 43501/1 А.М. Зинченко

(подпись)

Руководитель А.В. Мяснов

(подпись)

“ ” 2014 г.

Санкт - Петербург

2014

1. **Цель работы**

Изучить возможности триггеров, выполнить полученное задание

1. **Проведение работы**

*Создать триггер для автоматического заполнения ключевого поля*

Для начала необходимо генератор (последовательность):

CREATE SEQUENCE TICKET\_GEN;

ALTER SEQUENCE TICKET\_GEN RESTART WITH 0;

Создадим триггер для автоматического заполнения TICKETS\_ID в таблице TICKETS:

CREATE GENERATOR TICKET\_GEN;

SET TERM ^ ;

CREATE OR ALTER TRIGGER GEN\_TICK\_ID FOR TICKETS

ACTIVE BEFORE INSERT POSITION 0

AS

begin

IF (new.tickets\_id IS NULL) THEN

NEW.tickets\_id = GEN\_ID(ticket\_gen, 1);

end

^

SET TERM ; ^

*Создать триггер для контроля целостности данных в подчиненной таблице при удалении/изменении записей в главной таблице*

Создадим триггер, который удаляет билеты, входящие в отмененный заказ:

SET TERM ^ ;

CREATE OR ALTER TRIGGER DELETE\_TICKETS\_IN\_ORDER FOR "ORDER"

ACTIVE BEFORE DELETE POSITION 0

AS

begin

/\* Trigger text \*/

delete from tickets where tickets. order\_id=old.order\_id;

end

^

SET TERM ; ^

Можно было бы использовать другой подход – установить правило каскадного удаления:

alter table TICKETS

add constraint FK\_TICKETS\_3

foreign key ("ORDER")

references "ORDER"(ORDER\_ID)

on delete CASCADE

using index FK\_TICKETS\_3

*Создать триггер, который при добавлении новой позиции в маршрут или обновлении записей будет проверять корректность операции: не должно быть циклических частей маршрутов*

Создадим исключение:

CREATE EXCEPTION CYCLE\_ROUTE ' Маршрут не должен быть циклическим'

Скрипт создания триггера:

SET TERM ^ ;CREATE OR ALTER TRIGGER NOT\_CYCLIC\_ROUTE FOR ROUTESACTIVE BEFORE INSERT OR UPDATE POSITION 0asdeclare variable pred\_route\_id integer;begin pred\_route\_id=new.routes\_id; if(pred\_route\_id=new.nxt) then exception CYCLE\_ROUTE; select routes.routes\_id from routes where routes.nxt=:pred\_route\_id into :pred\_route\_id;while(row\_count=1)dobegin if(pred\_route\_id=new.nxt) then exception CYCLE\_ROUTE; select routes.routes\_id from routes where routes.nxt=:pred\_route\_id into :pred\_route\_id;endend^SET TERM ; ^

На самом деле есть более простое решение – установить поле nxt, как уникальное. При данном решении нужно придерживаться правила: каждому маршруту свои поля в таблице Routes

. alter table ROUTESadd constraint UNQ1\_ROUTES\_NXTunique (NXT)

*Создать триггер, который будет заполнять поле стоимости билета, вызывая процедуру для подсчета стоимости.*

Пусть стоимость билета вычисляется по формуле:

Скрипт создания триггера:

SET TERM ^ ;

CREATE OR ALTER TRIGGER INP\_COST FOR TICKETS

ACTIVE BEFORE INSERT OR UPDATE POSITION 0

AS

declare variable temp decimal(5,2);

begin

/\* Trigger text \*/

execute procedure compute\_cost new.st,new.vag,new.fin returning\_values temp;

new.cost=temp;

end

^

SET TERM ; ^

Процедура расчета цены:

SET TERM ^ ;

create or alter procedure COMPUTE\_COST (

ST\_POINT integer,

VAG integer,

FINISH\_POINT integer)

returns (

COST decimal(5,2))

as

declare variable TEMP integer;

declare variable TEMP2 decimal(3,2);

declare variable LEN decimal(15,2);

declare variable KOEF\_VAG decimal(15,2);

begin

if(st\_point is null or vag is null or finish\_point is null) then exit;

select vagons.vag\_type from vagons where :vag=vagons.vagons\_id

into :temp;

koef\_vag=CASE

WHEN temp=1 THEN 0.5

WHEN temp=2 THEN 1

when temp=3 then 2

ELSE null

END;

select routes.distance,routes.nxt from routes where routes.routes\_id=:st\_point

into :len, :temp;

while(temp~=:finish\_point)

do

begin

select routes.distance,routes.nxt from routes where routes.routes\_id=:temp

into :temp2, :temp;

len=len+temp2;

end

cost=round(maxvalue(34,:len\*1.93\*:koef\_vag),2);

suspend;

end^

SET TERM ; ^

GRANT SELECT ON VAGONS TO PROCEDURE COMPUTE\_COST;

GRANT SELECT ON ROUTES TO PROCEDURE COMPUTE\_COST;

GRANT EXECUTE ON PROCEDURE COMPUTE\_COST TO SYSDBA;

1. **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.